НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
(повна назва інституту/факультету)

Автоматизованих систем обробки інформації і управління
(повна назва кафедри)

«На правах рукопису»
УДК *004.02*

До захисту допущено:
В.о. завідувача кафедри
_____ Олександр ПАВЛОВ
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інженерія програмного забезпечення комп'ютеризованих систем»

зіспеціальності 121«Інженерія програмного забезпечення»

на тему: «Інженерія програмного забезпечення»

на тему «Інтелектуальна система для виявлення неполадок в процесі 3D друку за допомогою аналізу відео ряду в реальному часі»

Виконав:
студент VI курсу, групи ІП-92мп
Ядельський Ростислав Ігорович _____

Науковий керівник:
доцент, кандидат технічних наук,
Ліщук Катерина Ігорівна _____

Рецензент:
доцент, кандидат технічних наук,
Лісовиченко Олег Іванович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Автоматизованих систем обробки інформації і управління

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма - «Інженерія програмногозабезпечення комп'ютеризованих систем»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Олександр ПАВЛОВ
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Ядельському Ростиславу Ігоровичу

1. Тема дисертації *«Інтелектуальна система для виявлення неполадок в процесі 3D друку за допомогою аналізу відео ряду в реальному часі»*,науковий керівник дисертації*Ліщук Катерина Ігорівна, доцент, кандидат технічних наук*,затверджені наказом по університету від «26» жовтня 2020 р. № 3132-с

2. Термін подання студентом дисертації *18 грудня 2020р.*

3. Об'єкт дослідження:*моніторинг процесу 3D друку*

5. Перелік завдань, які потрібно розробити: *виконати аналіз основних неполадок в процесі друку, провести огляд наявних методів визначення неполадок, здійснити порівняльний аналіз зазначених методів, формалізувати задачу моніторингу процесу 3D друку, розробити метод виявлення неполадок, реалізувати програмне забезпечення, що реалізовуватиме розроблений метод, виконати аналіз результатів роботи створеного програмного забезпечення.*

6. Орієнтовний перелік графічного (ілюстративного) матеріалу*Архітектура нейронної мережі; Результат сегментації та генерації зображення.*

7. Орієнтовний перелік публікацій: *Виявлення неполадок в процесі 3ddруку за допомогоюавтоматизованоговізуальногомоніторингу.*

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Графічний	доц. Ліщук К.І.		

9. Дата видачі завдання:*1 вересня 2020р*

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	*Долідженняпредметної області*	*02.09.2020*	
2	*Дослідження літератури*	*10.09.2020*	
3	*Постановка та формулювання задач*	*15.09.2020*	
4	*Розробка методу виявлення неполадок*	*24.09.2020*	
5	*Аналіз технологічного стеку для реалізації*	*30.09.2020*	
6	*Реалізація програмного забезпечення*	*20.10.2020*	
7	*Дослідження ефективності*	*29.10.2020*	
8	*Розробка стартап-проекту*	*10.11.2020*	
9	*Оформлення документації*	*24.11.2020*	
10	*Подача роботи на попередній захист*	*27.11.2020*	
11	*Подача роботи на основний захист*	*18.12.2020*	

Студент Ядельський Ростислав Ігорович

Науковий керівник Ліщук Катерина Ігорівна

РЕФЕРАТ

Актуальність теми. Протягом останніх років настільні 3D-принтери стали достатньо доступними та розповсюдженими. Однак, попри свою поширеність, експлуатація таких 3D принтерів є достатньо складною так як у процесі друку з великою вірогідністю може виникнути несправність. Залежно від розміру, складності моделі та обраної роздільної здатності під час генерації інструкцій для принтеру, процес 3D-друку може зайняти від пари годин до декількох днів. Це означає, що машина іноді працюватиме великий проміжок часу без людського контролю. За цей час багато речей може піти не так. Наприклад, сопло, через яке видавлюється матеріал, може бути закупоритись, або виріб може відлипнути від друкарського столу. Це може привести до марнування великої кількості матеріалу, значних часових затрати та до пошкодження принтеру. Якщо проблему виявлено на ранньому етапі, багатьох наслідків можна уникнути. Зараз єдиним способом виявлення цих помилок є ручна перевірка процесу друку людиною. Автоматизація цього процесу є необхідною для розширення сфери застосування 3D друку та полегшення роботи з цією технологією.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» згідно з планом науково-дослідницьких робіт кафедри автоматизованих систем обробки інформації та управління.

Метою цієї роботи є розроблення архітектури програмних засобів для автоматизації моніторингу процесу 3D друку, що в свою чергу дозволить спростити та зробитиефективнішою експлуатацію 3D принтерів. Також це дозволить мінімізувати необхідність присутності оператора 3D принтера, що більше не буде змушений періодично перевіряти стан об’єкта друку.

Для досягнення мети необхідно виконати наступні **завдання**:

- виконати аналіз основних неполадок в процесі друку;
- провести огляд наявних методів визначення неполадок;

- здійснити порівняльний аналіз зазначених методів;
- формалізувати задачу моніторингу процесу 3D друку;
- розробити метод виявлення неполадок;
- реалізувати програмне забезпечення, що реалізовуватиме розроблений метод;
- виконати аналіз результатів роботи створеного програмного забезпечення.

Об’єкт дослідження– моніторинг процесу 3D друку.

Предмет дослідження – програмні методи та системні рішення для виявлення неполадок в процесі 3D друку.

Методи дослідження, застосовані в даній роботі базуються на підходах комп’ютерного зору, комп’ютерної графіки та машинного навчання.

Наукова новизна одержаних результатів полягає в унікальності розробленої системи, аналогів якої немає, та узастосуванніінноваційного методу виявлення неполадок 3Dдруку, що базується на поєднання методів комп’ютерного зору, комп’ютерної графіки та машинного навчання, що також відрізняється врахуванням властивостей моделі об’єкта, що друкується.

Практичне значення одержаних результатів зумовлена реалізацією розробленого методу в системі для автоматичного моніторингу процесу 3D друку, яка може застосовуватись для спрощення експлуатації 3D принтерів.

Публікації. Матеріали роботи опубліковані в збірнику тез IV всеукраїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020).

3D ДРУК, МОНІТОРИНГ, КОМП’ЮТЕРНИЙ ЗІР, КОМП’ЮТЕРНА ГРАФІКА, НЕЙРОННІ МЕРЕЖІ, ВИЯВЛЕННЯ НЕПОЛАДОК

ABSTRACT

Actuality. In recent years, desktop 3D printers have become available and widespread. However, due to its widespread popularity, the use of such 3D printers is quite complex, as the printing process is more likely to malfunction. Depending on the size, complexity of the models and image resolution when generating printing instructions, the 3D printing process may depend on the pair. hours to several days. This means that the machine sometimes runs for a long time without human control. During this time, many things can go wrong. For example, the nozzle through which the material is removed may be purchased, or the solution may peel off the printing table. This can result in marking of large amounts of materials, significant time delays, and damage to the printer. If the problem is detected early, many consequences can be changed. Currently, the only way to detect these errors is to manually check the human printing process. Automation of this process is necessary to expand the scope of 3D-printing and selection of work on this technology.

Relationship with working with scientific programs, plans, topics. The work was performed at the Department of Automated Information Processing and Control Systems of the National Technical University of Ukraine "Kyiv Polytechnic Institute. Igor Sikorsky" as part of the topic "Intelligent system for detecting problems in the process of 3D printing using real-time video series analysis".

The aim of this work is to automate the monitoring of the 3D printing process to simplify and reduce resources for the operation of 3D printers. As a result, you can significantly reduce the time to troubleshoot, which in turn will reduce the cost of wasted material and printer maintenance. It will also minimize the need for a 3D printer operator, who will no longer have to periodically check the condition of the print object, which will reduce both time and money to manufacture the product.

To achieve this goal, you must perform the following **tasks**:

- perform an analysis of major problems in the printing process;
- review existing troubleshooting methods;
- carry out a comparative analysis of these methods;
- formalize the task of monitoring the 3D printing process;

- develop a method of detecting problems;
- implement software that will implement the developed method;
- perform analysis of the results of the created software.

The object of research is monitoring of the 3D printing process.

Subjects of research are methods for detecting problems in the process of 3D printing.

The research methods used in this paper are based on the approaches of computer vision, computer graphics and machine learning.

The scientific novelty of the results is the use of a unique combination of computer vision, computer graphics and machine learning to detect problems in the process of 3D printing, which also takes into account the properties of the model of the object being printed.

The practical significance of the obtained results is due to the implementation of the developed method in the system for automatic monitoring of the 3D printing process, which can be used to simplify the operation of 3D printers.

Publications. Materials of the work are published in the collection of abstracts of the IV All-Ukrainian scientific-practical conference of young scientists and students "Information systems and management technologies" (ISTU-2020).

3D PRINTING, MONITORING, COMPUTER VISION, COMPUTER GRAPHICS, NEURAL NETWORKS, FAILURE DETECTION

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

FDM – Fused Deposition Modeling, моделювання методом пошарового наплавлення

FFF – Fused filament fabrication, синонім до FDM, моделювання методом розплавлення нитки матеріалу

CNN – Convolutional Neural Network, згорткова нейронна мережа

HSV – колірна модель, заснована на трьох характеристиках кольору: колірному тоні (Hue), насиченості (Saturation) і значенні кольору (Value), який також називають яскравістю (Brightness).

REST – Representational State Transfer, підхід до архітектури мережевих протоколів, які надають доступ до інформаційних ресурсів.

JSON – JavaScript Object Notation, текстовий формат обміну даними між комп'ютерами.

G-CODE – мова програмування для пристроїв з числовим програмним керуванням.

Слайсер – програмне забезпечення для генерації G-CODE із заданої 3D моделі.

Raspberry Pi – одноплатний комп'ютер, розроблений британським фондом Raspberry Pi Foundation.

MJPEG – назва групи стандартів кодування відео, в яких кожен кадр цифрової відео послідовності незалежно закодований за алгоритмом JPEG.

ВСТУП

3D-принтери стали достатньо популярними в останнє десятиліття, так як вони дозволяють швидко та не дорого створювати прототипи виробів всіх можливих форм та розмірів. За допомогою технології адитивного виробництва можна створювати фізичні об’єкти на основі даних 3D-моделі шляхом пошарового додавання матеріалу. Окрім професійного використання для створення прототипів та виробництва невеликих обсягів, вони набувають широкого розповсюдження серед кінцевих споживачів, які використовують 3D-принтери для хобі та у господарстві. Найпоширенішим типом 3D-принтерів споживчого класу є принтери, що працюють за принципом моделювання методом наплавлення (FusedDepositionModelling - FDM, також FFF). Ця робота зосереджена на механізмах FDM через їх широке поширення та велику кількість наявних проблем, таких як низька точність та відмови.

Процес FDM друку на теперішньому етапі розвитку технологій важко назвати надійним і в залежності від виробника та моделі принтера проблеми можуть виникати з різною регулярністю. Пошкодження можуть статися через неправильне вирівнювання поверхні для друку, друкуючої головки, пробуксовки двигунів, викривлення друкованого матеріалу, відсутність зчеплення або інші причини. Ціллюданого дослідження є створення середовища, в якому ці збої можуть бути виявлені автоматично. Безпосередній нагляд зі сторони людини перешкоджає рекомендованому розміщенню принтерів FDM в окремих приміщеннях подалі від користувача через проблеми з вентиляцією. Неможливість нагляду за процесом друку призводить до пізнього або пропущеного виявлення неполадок.

Для цього необхідно розробити систему, що складається з механізму виявлення помилок на основі камери, який забезпечує веб-інтерфейс для віддаленого контролю та раннього виявлення неполадок. Раннє виявлення неполадок може призвести до зменшення часу, витраченого на невдалі вироби, зменшення витрачання матеріалу, а в деяких випадках і пошкодження предметів. Також, при подальшому покращенні дану технологію можна буде

використовувати для повної автоматизації процесу 3D друку, що зараз неможливо саме через відсутність надійної системи моніторингу.

Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» згідно з планом науково-дослідницьких робіт кафедри автоматизованих систем обробки інформації та управління.

Метою цієї роботи є розроблення архітектури програмних засобів для автоматизації моніторингу процесу 3D друку, що в свою чергу дозволить спростити та зробитиефективнішою експлуатацію 3D принтерів. Також це дозволить мінімізувати необхідність присутності оператора 3D принтера, що більше не буде змушений періодично перевіряти стан об’єкта друку.

Для досягнення мети необхідно виконати наступні **завдання**:

- виконати аналіз основних неполадок в процесі друку;
- провести огляд наявних методів визначення неполадок;
- здійснити порівняльний аналіз зазначених методів;
- формалізувати задачу моніторингу процесу 3D друку;
- розробити метод виявлення неполадок;
- реалізувати програмне забезпечення, що реалізовуватиме розроблений метод;
- виконати аналіз результатів роботи створеного програмного забезпечення.

Об’єкт дослідження– моніторинг процесу 3D друку.

Предмет дослідження – програмні методи та системні рішення для виявлення неполадок в процесі 3D друку.

Методи дослідження, застосовані в даній роботі базуються на підходах комп’ютерного зору, комп’ютерної графіки та машинного навчання.

Наукова новизна одержаних результатів полягає в унікальності розробленої системи, аналогів якої немає, та узастосуванніінноваційного методу виявлення неполадок 3Dдруку, що базується на поєднання методів комп’ютерного зору,

комп’ютерної графіки та машинного навчання, що також відрізняється врахуванням властивостей моделі об’єкта, що друкується.

Практичне значення одержаних результатів зумовлена реалізацією розробленого методу в системі для автоматичного моніторингу процесу 3D друку, яка може застосовуватись для спрощення експлуатації 3D принтерів.

Матеріали роботи опубліковані в збірнику тез IV всеукраїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020).

1 АНАЛІЗ НЕПОЛАДОК В ПРОЦЕСІ 3D ДРУКУ

1.1 Принцип роботи FDM 3D принтерів

Для того щоб виявляти можливі проблеми в процесі 3D друку слід розглянути як працює моделювання методом наплавлення.

Процес виготовлення виробу починається з 3D моделі. Люди самостійно створюють віртуальні тривимірні моделі потрібної форми за допомогою спеціальних програм для моделювання та проектування. Форма віртуальної моделі задається точками в тривимірному просторі. У файлі з моделлю записані координати кожної вершини цієї фігури.

Спеціальна програма 3D принтера, яка називається слайсер (від англійського “slice” – різати), розрізає тривимірні моделі на окремі плоскі шари, які потім будуть надруковані один за іншим. У програмі вказують швидкість і точність друку, температуру та інші параметри. Налаштування передаються спеціальними командами у форматі GCODE, які виконує 3D принтер.

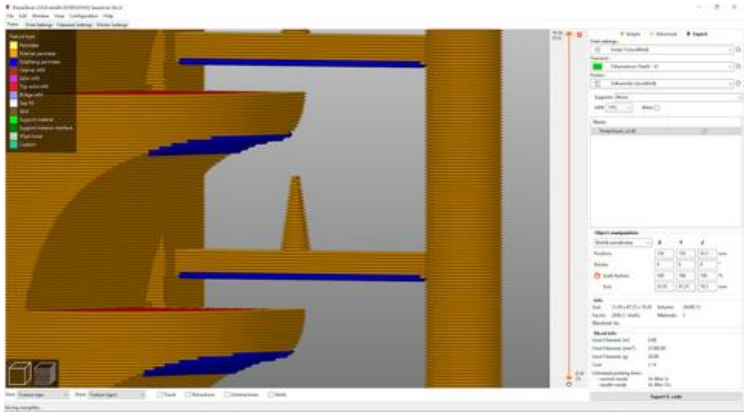


Рисунок 1.1 – Інтерфейс слайсера

```
233 G1 X113.842 Y157.651 E0.0535 F1500
234 G1 X112.944 Y157.176 E0.0912
235 G1 X112.012 Y156.531 E0.1333
236 G1 X111.314 Y155.927 E0.1676
237 G1 X110.423 Y154.970 E0.2161
238 G1 X109.736 Y153.984 E0.2607
239 G1 X109.191 Y152.906 E0.3056
240 G1 X108.944 Y152.310 E0.3295
241 G1 X93.634 Y105.604 E2.1543
242 G1 X88.116 Y103.158 E2.3784
243 G1 X87.560 Y102.860 E2.4019
244 G1 X86.573 Y102.234 E2.4452
245 G1 X85.865 Y101.672 E2.4788
246 G1 X84.967 Y100.789 E2.5256
247 G1 X84.444 Y100.165 E2.5558
248 G1 X83.801 Y99.230 E2.5979
249 G1 X83.405 Y98.519 E2.6281
```

Рисунок 1.2 – Приклад файлу GCODE

Після підготовки 3D моделі запускають 3D принтер, завантажують необхідний тип пластика і приступають до друку. Команди GCODE передаються принтеру або безпосередньо з комп'ютера через звичайний USB кабель, з можливістю коригування процесу в реальному часі, або створивши спеціальний * GCODE файл, в якому буде весь необхідний перелік команд, що дозволяє принтеру підготуватися до друку, надрукувати модель і завершити друк самостійно і практично без втручання з боку.

Моделювання методом наплавлення або FusedDepositionModelling (FDM) було винайдено ученим на ім'я Скотт Крамп через кілька років після того, як Чак Хулл винайшов лазерний 3D друк. Крамп намагався дістати вигоду з розробки і в 1990 році заснував компанію Stratasys, яка запатентувала цю технологію під брендом FFF. З цієї причини сама найпопулярніша технологія 3D друку FDM часто згадується як FusedFilamentFabrication (FFF).[1]

Принцип, за яким працює ця технологія, досить простий. Саме тому 95% всіх настільних 3D принтерів використовують FDM або FFF. Пластик, такий як PLA або ABS, у формі нитки подається в екструдер, серце принтера. У екструдері пластикова нитка розігрівається і переходить у рідкий стан. Механічні частини принтера слідують командам з файлу GCODE і переносять екструдер в потрібне положення строго за вказаними координатами. Коли екструдер досягає заданої позиції, пластик виходить з гарячого сопла, приклеюючись до столу принтера або до минулихшарів.

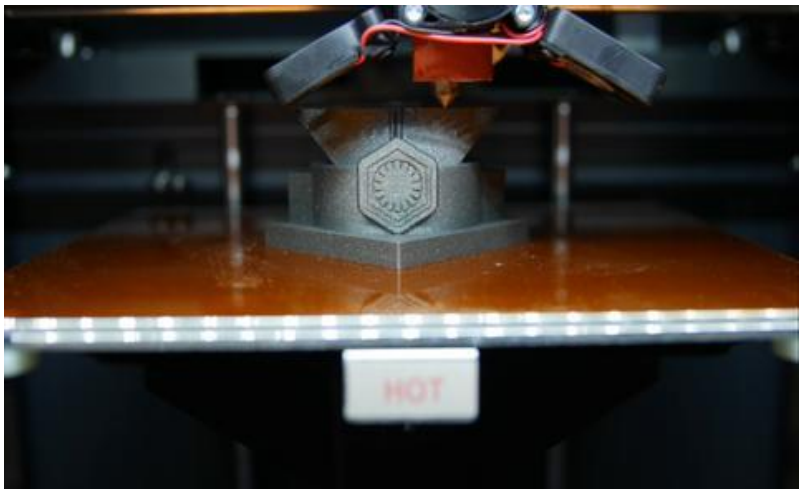


Рисунок 1.3 – Процес 3D-друку

Через секунди після друку пластик твердне і надрукована модель стає жорсткою. Важливо стежити за правильною температурою пластика, основи принтера і повітря в приміщенні, інакше через нерівномірний остигання в деталі можуть накопичуватися внутрішні напруження, що призводить до деформації або втрати міцності.

Оскільки 3D принтер друкує моделі пошарово, для кожного нового шару необхідна опора на попередній, інакше новий шар пластику виявляється в повітрі і прогинається. Іноді форма моделей така, що нависають частини моделі не мають достатньої опори, в таких випадках програма принтера автоматично додає конструкцію підтримки з матеріалу, який розчиняється в спеціальній рідині. Після друку конструкція підтримки видаляється.

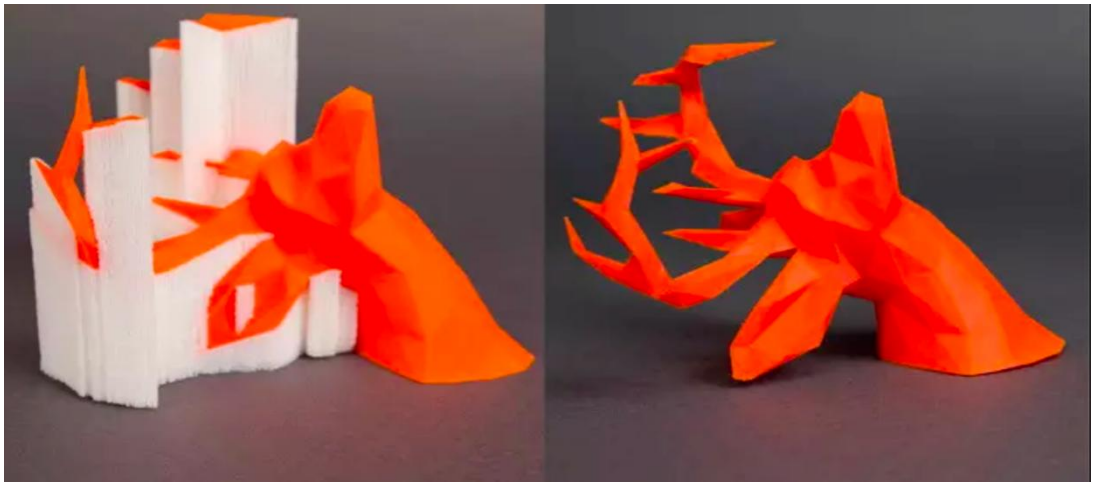


Рисунок 1.4 – Підтримки

Залежно від необхідної міцності виробу, ми встановлюємо різну ступінь заповнення внутрішнього простору моделей. Починаючи від заповнення 0%, коли принтер друкує тільки оболонку моделі, і закінчуючи заповненням 100%, коли деталь повністю заповнена пластиком. Для прототипів раціонально використовувати заповнення 20%, зі збільшенням заповнення зростає і вага виробу, разом з ним вартість 3D друку, зі зменшенням деталь стає занадто крихкою.

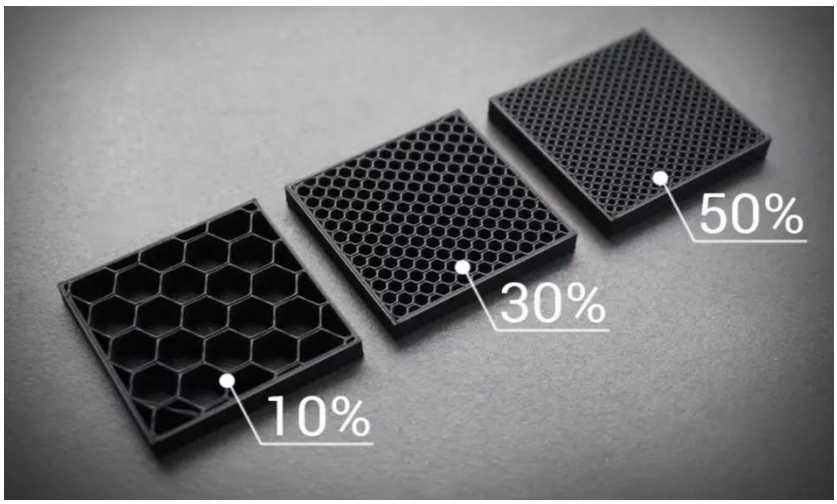


Рисунок 1.5 – Внутрішнє заповнення моделей

Товщина стінок деталі регулюється окремо. Для невеликих деталей вона зазвичай становить 1 міліметр.

Як описано вище, в процесі 3D друку задіяна досить велика кількість процесів, коректність проведення яких не відслідковується автоматично, тому часто виникають різного виду неполадки. Для того, щоб їх успішно виявляти слід описати причини їх появи та їх ознаки. Основні типи неполадок перелічені далі.

1.2 Відрив об’єкта від поверхні для друку

Найпоширеніша помилка і найпростіша у виявленні - це від'єднання об'єкта від поверхні для друку. Об’єкт не повинен від’єднуватися для успішного друку. На рисунку 1.6 ця ситуація представлена двома ескізами. Смугастий прямокутник із закругленими кутами представляє об’єкт для друку та прикріплюється до поверні друку (представлена товстою чорною смугою під об’єктом). У друку без помилок цей предмет міцно прикріплений до друкарської поверхні і не рухається в будь-якому напрямку відносно поверхні. У даному прикладі принтер має друкарський стіл, який має один ступінь свободи в напрямку Z (висота) і друкуючу головку з двома ступенями свободи в напрямку X та Y (ліворуч / праворуч і вперед / назад). Інші конфігурації принтерів можуть потребувати регулювання, оскільки там друкарський стіл може рухатися у двох вимірах.[2]

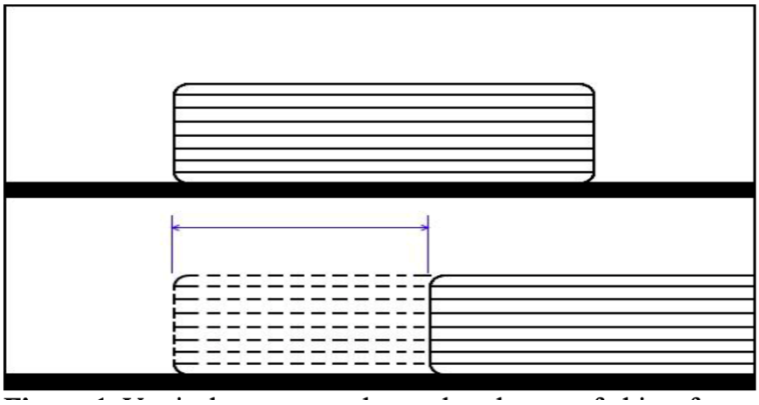


Рисунок 1.6 – Ескіз від’єднання об’єкта від друкарського столу

На ескізі відрив видно, оскільки об’єкт перемістився вправо, а початкове положення позначено пунктирними лініями. Зміщення між початковим та фактичним положенням називається помилкою переміщення. Цей тип / клас помилки може мати місце:

– коли температура друкарського ложа коливається, а предмет охолоджується нерівномірно;

– коли друкарський лоток не відрегульований та відкалібрований, а тому відстань між соплом та друкарською поверхнею змінюється;

– через до вібрації або удару, що від'єднує предмет.

В результаті відриву часто можна побачити велику кількість заплутаних згустків та ниток пластику, як зображено на рисунку 1.7.

Рисунок 1.7 – Від’єднання об’єкта від друкарського столу

1.3 Зупинка постачання пластику

Другою найбільш частою проблемою є зупинка постачання пластику в процесі друку. При цьому типі помилки термопластик не протікає через сопло друкуючої головки. Друкувальна головка рухається вздовж попередньо визначеного шляху інструменту без видавлювання нитки пластику. Висота об’єкта залишається незмінною, а об’єкт рухається вздовж осі Z вниз. Об’єкт не завершений і не буде завершений. Якщо потік матеріалу буде перервано лише тимчасово, об’єкт буде надрукований з дефектом, оскільки один або кілька шарів матеріалу відсутні. Якщо потік матеріалу перерваний лише ненадовго, об’єкт може бути повністю надрукований лише з незначними дефектами, оскільки шари можуть компенсувати незначну частину нижнього шару.

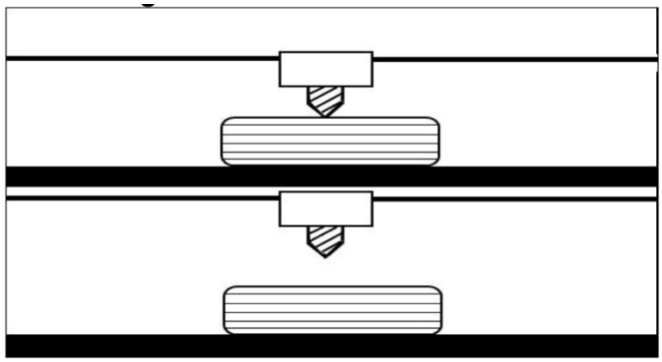


Рисунок 1.9 – Ескіз зупинки постачання пластику

На рисунку 1.9 зображено цей тип помилки. Об'єкт відображається у вигляді смугастого прямокутника із закругленими кутами, друкарський лоток - у вигляді товстої чорної лінії під об'єктом друку, а друкуюча головка або екструдер - у вигляді білого прямокутника із смугастим п'ятикутником внизу. У верхній частині ескізу показано випадок, що об’єкт друкується нормально без помилок, а екструдер прилягає до об’єкта. [4] У нижній частині показано випадок, коли потік матеріалу зупинився, екструдер більше не торкається предмета і рухається вздовж його інструментальної траєкторії. Цей клас помилок може виникнути через:

- засмічення сопла чи екструдера;
- бульбашки повітря в екструдері;
- заклинений матеріал (пруток матеріалу розірваний або двигун екструдера не може схопити пруток для транспортування в сопло).

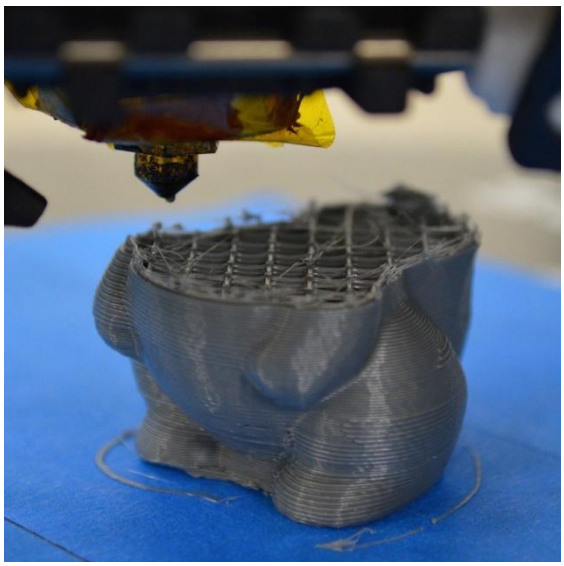


Рисунок 1.10 - Зупинка постачання пластику

1.4 Зсув шарів

Ще одною розповсюдженою несправністю є зсув шарів в процесі друку. Приклад такої несправності наведений на рисунку 1.11. Зверху зображена нормальна укладка матеріалу, а знизу – при зсуві шарів.

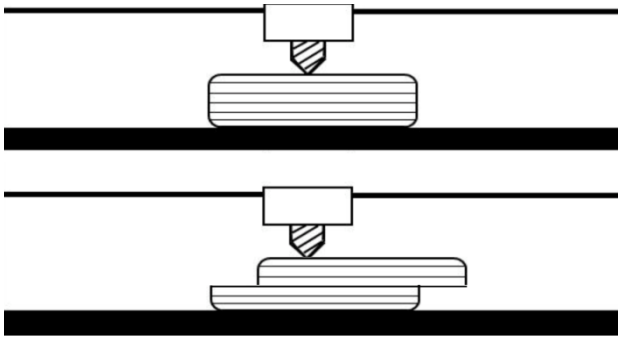


Рисунок 1.11 – Ескіз зсув шарів

Ця проблема виникає при несправності системи принтера, що відповідає за рух екструдера, наприклад заїдання мотора. Приклад такої несправності наведений на рисунку 1.12.

Рисунок 1.12 – Зсув шарів

1.5 Постановка завдання дослідження

Метою цієї роботи є розроблення архітектури програмних засобів для автоматизації моніторингу процесу 3D друку, що в свою чергу дозволить спростити та зробитиефективнішою експлуатацію 3D принтерів. Також це дозволить мінімізувати необхідність присутності оператора 3D принтера, що більше не буде змушений періодично перевіряти стан об’єкта друку.

Для досягнення мети необхідно виконати наступні завдання:

- виконати аналіз основних неполадок в процесі друку;
- провести огляд наявних методів визначення неполадок;
- здійснити порівняльний аналіз зазначених методів;
- формалізувати задачу моніторингу процесу 3D друку;
- розробити метод виявлення неполадок;
- реалізувати програмне забезпечення, що реалізовуватиме розроблений метод;
- виконати аналіз результатів роботи створеного програмного забезпечення.

Висновокрозділу

В цьому розділі було розглянуто принципуроботи FDM 3D друку від створення 3D моделі, до власне отримання матеріального об’єкта та зроблено огляд основних неполадок, що виникають у процесі друку. Було виділено та

описано причини виникнення трьох типів неполадок, а саме відрив об’єкта від поверхні для друку, зупинка подачі матеріалу та зсув шарів. Завдання цієї роботи полягає у їх автоматичному виявленні.

2 МЕТОДИ ВИЯВЛЕННЯ НЕПОЛАДОК ДРУКУ ЗА ДОПОМОГОЮ АНАЛІЗУ ВІЗУАЛЬНОЇ ІНФОРМАЦІЇ

2.1 Огляд системи

Оскільки конструкція 3D принтерів не дозволяє реалізувати механічний контроль якості друку, єдиним доступним способом моніторингу є візуальний контроль з допомогою відео камер. Для підвищення якості моніторингу необхідно використовувати кілька камер, що дозволить використовувати алгоритм триангуляції для розпізнавання позиції об’єкта, що друкується, однак це потребує точного позиціонування системи камер, чого важко досягнути при експлуатації настільних 3D принтерів, тому для виявлення неполадок методом, описаним у даній статті використовується єдина камера та джерело світла, що встановлені статично та напрямлені на поверхню для друку, як зображено на рисунку 2.1. [3] Кольорова камера знімає зображення, освітленої моделі після додавання кожного нового шару матеріалу. Для того, щоб спростити аналіз зображень, програму управління принтером модифіковано таким чином, щоб при завершенні друку кожного шару друкуюча голівка переміщалась у заздалегідь задану точку у горизонтальній площині, щоб не перекривати модель на зображенні, після чого власне і відбуватиметься створення фото. Це дозволяє позбутись необхідності постійного відслідковування положення друкуючої голівки та нівелювати вплив її руху на аналіз. Таким чином єдина різниця між зображеннями двох сусідніх шарів буде зумовлена власне викладеним на останньому шарі матеріалом.

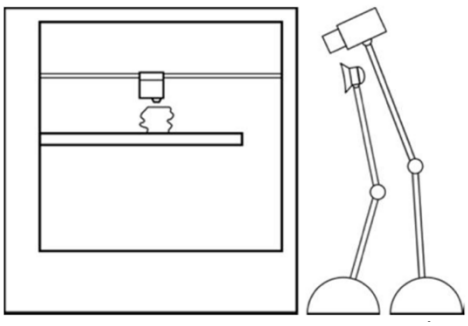


Рисунок2.1 – Встановлення камери та джерела світла для спостереженням за процесом друку

Для виявлення неполадок у даній статті розглянуто наступні методи:

– використання класичного підходу комп’ютерного зору та виявлення помилок базуючись на аналіз різниці між зображеннями шарів моделі;

– комбінація підходів класичного комп’ютерного зору та аналізу комп’ютерної моделі виробу, що друкується для виявлення різниці між реальними та симульованими відхиленнямиу зображеннях;

– аналіз зображень з використанням згорткових нейронних мереж для виявлення характерних артефактів, що сигналізують про неполадку в процесі друку.

2.2 Метод виявлення неполадок друку з використанням класичних підходів комп’ютерного зору

Для коректного виявлення несправності на зображенні необхідно відділити об’єкт від фону зображення. При сегментації об’єкту із захопленого зображення використовується колір матеріалу для друку. Оскільки цей підхід спрямований на недорогі споживчі принтери, матеріалом, як очікується, є монохромна пластмаса PLA. Камера робить кольорові зображення за допомогою техніки фільтра Байєра. Колір кожного пікселя представлений трьома інтенсивностями, по одній для кожного з червоного, зеленого та синього кольорів (RGB). У поєднанні ці три кольори можуть імітувати більшість сприйманих людиною кольорів. Хоча це добре для людей, це кольорове зображення не є ідеальним для алгоритмічної сегментації кольорових предметів, тому для сегментації проводиться перетворення кольорового простору з RGB на відтінок, насиченість і значення (HSV). Простір HSV представляє колір як набір з трьох значень: чистий колір як кут від 0 ° до 360° на крузі кольорів і насиченість та інтенсивність як значення від 0 до 1.[6]

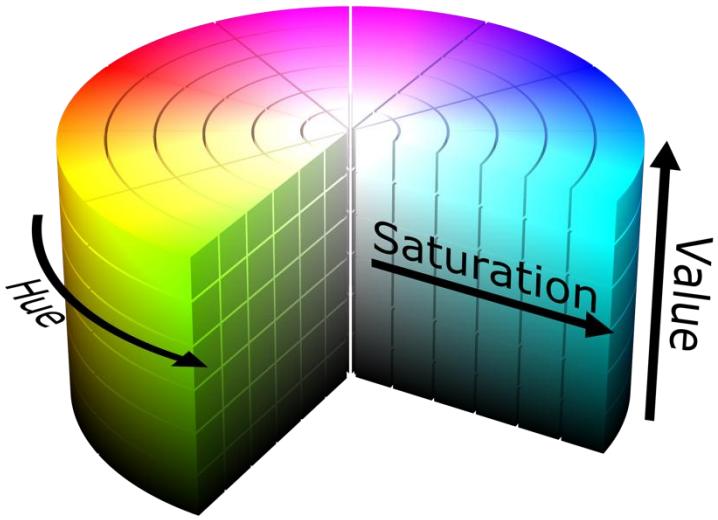


Рисунок 2.2 – Кольоровий простір HSV

Маючи піксель зі значенням у просторі RGB зі значеннями $r, g, b \in [0, 255]$ спершу відбувається їх нормалізація:

$$r' = r \,/\, 255$$
$$g' = g \,/\, 255$$
$$b' = b \,/\, 255$$

Після цього визначається максимальна та мінімальна компонента і вираховується їх різниця:

$$C_{max} = \max(r', g', b')$$
$$C_{min} = \min(r', g', b')$$
$$\Delta = C_{max} - C_{min}$$

Ці значення використовуються для обчислення компонент в представленні HSV:

$$h = \begin{cases} 60 * \mathrm{mod}\left(\frac{g-b}{\Delta}, 6\right) & C_{max} = r' \\ 60 * \left(\frac{b-r}{\Delta} + 2\right) & C_{max} = g' \\ 60 * \left(\frac{r-g}{\Delta} + 4\right) & C_{max} = b' \end{cases}$$

$$s = \begin{cases} 0 & C_{max} = 0 \\ \frac{\Delta}{C_{max}} & C_{max} > 0 \end{cases}$$

$$v = C_{max}$$

Після того, як зняті зображення переведені в кольоровий простір HSV пластик сегментується по ознаці належності до діапазону $H_t = [h_{min}, h_{max}]$, що відповідає діапазону для заданого матеріалу. Крім того, також відкидуються пікселі зі значенням компонент s та v менших за порогові значення T_s та T_v відповідно для видалення темних пікселів, що не несуть корисної інформації. Таким чином, для створення сегментаційної маски використовується наступна формула:

$$S = \begin{cases} 1 & h \in H_t, s > T_s, v > T_v \\ 0 & \text{в іншому випадку} \end{cases}$$

Коли сегментація проведена, для виявлення несправності в процесі друку визначається середньокватичневідхилення між зображеннями двох сусідніх шарів під номерами *l* та *l-1*:

$$RMSE = \sqrt{\frac{\sum_{x=1}^{X_{max}} \sum_{y=1}^{Y_{max}} (S_{x,y,l-1} - S_{x,y,l})^2}{X_{max} * Y_{max}}}$$

Якщо дане значення більше за зазначене порогове значення E_t, система сигналізує про ймовірну несправність друку:

$$Error = \begin{cases} 1 \; RMSE > E_t \\ 0 \; RMSE \leq E_t \end{cases}$$

2.3 Метод виявлення неполадок друку з використанням комп'ютерного зору та аналізу комп'ютерної моделі виробу

При використанні методу, описаного у минулому пункті, найскладнішою задачею є правильний підбір граничного значення E_t, так як оптимальне його значення залежить від очікуваної кількості матеріалу, що викладається на кожному шарі об'єкта та положення камери. Таким чином для різних моделей, положень камери, і навіть для різних шарів одної моделі, це значення повинно змінюватись. Для того, щоб автоматизувати обрахунок оптимального значення E_t можна використовувати комп'ютерну модель виробу, що друкується. Нехай

товщина одного шару моделі H_l, тоді висоту частини моделі, що вже надрукована на шарі l можна визначити за наступною формулою:

$$h_l = H_l * l$$

Відітнувши верхню частину моделі горизонтальною площиною, що знаходиться на висоті h_l отримуємо комп'ютерну модель виробу, що в даний момент знаходиться на поверхні для друку, яка використовується для створення комп'ютерно генерованого зображення виробу. Щоб створити зображення слід врахувати взаємне положення між камерою та об'єктом після чого до моделі застосовується перспективна проекція:

$$S = \frac{1}{\tan\left(\frac{fov}{2} * \frac{\pi}{180}\right)}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ w' \end{pmatrix} = \begin{bmatrix} S & 0 & 0 & 0 \\ 0 & S & 0 & 0 \\ 0 & 0 & -\frac{f}{(f-n)} & -1 \\ 0 & 0 & -\frac{f*n}{(f-n)} & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$u = \frac{x'}{z'} \quad v = \frac{y'}{z'}$$

У формулах наведених вище fov це кут огляду камери, f та n – це відстані до дальньої та ближньої обтинаючоїплощин відповідно, що необхідні для коректної роботи процесу рендерингу зображення з використанням програмного комплексу OpenGL. Результат застосування цього перетворення показано на рисунку 2.3. Оскільки генероване зображення буде застосовуватися як бінарна маска, то для його генерації не застосовується модель освітлення. Для кращого позиціонування моделі у просторі відносно камери використовується порівняння реального та згенерованого зображення виробу.

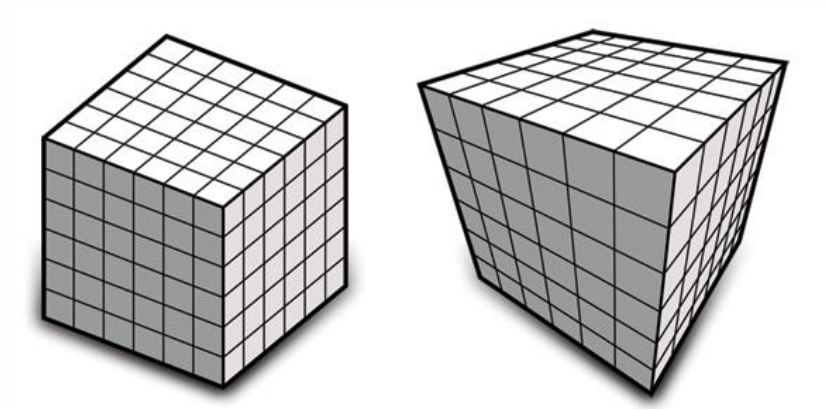


Рисунок 2.3 – Результат застосування перспективної проекції на 3D модель

Після отримання комп’ютерно генерованого зображення виробу, що друкується на шарі l визначається прогнозованевідхиленнята порівнюється з відхиленням на реальних зображеннях. Якщо вони відрізняються більш ніж заздалегідь заданий допуск ε, то система сигналізує про помилку друку:

$$E = \frac{\left|RMSE_{cam} - RMSE_{gen}\right|}{RMSE_{gen}}$$

$$Error = \begin{cases} 1 & E > \varepsilon \\ 0 & E \leq \varepsilon \end{cases}$$

Крім описаного вище методу, за умови точного позиціонування камери, є можливість визначення відхилення між згенерованим та реальним фото, що дозволить виявляти неточності в геометрії виробу.

2.4 Метод виявлення неполадок з допомогоюзгорткових нейронних мереж

При виникненні неполадок у процесі друку часто подача матеріалу не припиняється, натомість він видавлюється просто в повітря утворюючи заплутані клубки характерного вигляду. Ідея цього методу полягає у виявленні таких клубків з допомогою згорткової нейронної мережі (ЗНМ). Згорткові нейронні мережі (англійською CNN) в машинному навчанні — це клас штучнихглибинних нейронних мереж прямого поширення, який успішно застосовувався для аналізу зображень. ЗНМ складається з шарів входу та виходу, а також із декількох прихованих шарів. На відміну від звичайних нейронних мереж, шари згорткових нейронних мереж є багатовимірними. Нейрони одного шару

пов’язані тільки з певною областю наступного шару. Перший вхідний шар мережі має розмірність $w \times h \times d$, де w — ширина зображення, h — висота, d — кількість каналів кольору. Кожен шар такої мережі трансформує отриманий вхід, використовуючи диференційовану функцію. В згортковій нейронній мережі в основному використовуються три типи шарів: згорткові, агрегуючі таповнозв’язні.

Згортковий шар є основним будівельним блоком ЗНМ. Параметри шару складаються з набору фільтрів (або ядер). Для обчислення значення функції активації у кожній точці значення кожного елемента фільтра перемножується на значення відповідного елемента вхідної матриці, після чого обчислюється суму відповідних добутків і вже з них формується матриця активації, яку ще називають картою активації.

При роботі з високорозмірними входами, такими як зображення, недоцільно сполучати нейрони одного шару до всіх нейронів у попередньому шару, оскільки така архітектура мережі не враховує просторову структуру даних. Кожен нейрон згорткової нейронної мережі пов’язаний лише з невеликою областю вхідних даних з попереднього шару.

Ступінь цього зв’язку - це гіперпараметр, який називається рецептивним полем нейрона. З'єднання локальні в просторі (по ширині та висоті), але завжди простягаються по всій глибині вхідного об'єму даних. Така архітектура гарантує, що вивчені фільтри забезпечують найсильніший відгук і їх активація не нівелюється наступними шарами.

Для навчання такої нейронної мережі необхідна велика кількість зображень, на яких наявні описані вище клубки матеріалу. Оскільки у вільному доступі таких даних немає, вони були створені штучно, з допомогою накладання на відео процесу друку випадково генерованих клубків матеріалу. Для генерації випадковим чином обирається кількість точок ламаної N, після чого координати кожної точки тривимірної ламаної обираються випадково, однак з умовою їх розміщення у сфері радіусом R. Отримана тривимірну ламана візуалізується у двовимірне зображення з урахуванням тіней від джерела світла. Згенероване

зображення накладається на кадр відео з реальної зйомки процесу друку і використовується для навчання нейронної мережі.

Висновок розділу

В даному розділі було описано загальну архітектуру системи таметоди виявлення основних неполадок в процесі 3D друку. Кожен з методів детально проаналізований, з урахуванням його переваг та недоліків. В результаті виявлено, що для досягнення найкращого результату слід використовувати комбінований підхід з використанням комп’ютерного зору, аналізу моделі об’єкта, та виявлення клубків матеріалу з допомогою нейронних мереж.

Опис побудови системи, що реалізує розроблений метод наводиться у наступному розділі.

3 АРХІТЕКТУРА ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Програмне забезпечення для управління та взаємодії з принтером

Оскільки система повинна тісно інтегруватись з програмою управління 3D принтером для управління процесом друку необхідно велику увагу приділити вибору оптимальної прошивки принтера. На даний момент найбільш актуальними є наступні рішення:

- Marlin;
- Klipper;
- Repetier;
- RepRap.

3.1.1 Marlin

MarlinFirmware працює на головній платі 3D-принтера, керуючи всіма діями машини в режимі реального часу. Він координує нагрівачі, крокові двигуни, датчики, ліхтарі, РК-дисплей, кнопки та все інше, що бере участь у процесі 3D-друку. Мова управління для Marlin є похідною від G-коду. Команди G-коду говорять машині робити такі прості речі, як «встановити нагрівач 1 на 180 ° температуру» або «перейти до XY зі швидкістю F». Коли Marlinотримує команди руху, він додає їх до черги руху, яка виконується в отриманому порядку. «Крокове переривання» обробляє чергу, перетворюючи лінійні рухи в електронні імпульси з точним синхронізацією до крокових двигунів.[9]

Навіть на невеликих швидкостях Marlinповинен генерувати тисячі крокових імпульсів щосекунди. (наприклад, 80 кроків на мм * 50 мм / с = 4000 кроків на секунду). Нагрівачами та датчиками управляється секундними перериваннями, які виконується набагато повільніше, в той час як основний цикл обробляє команди обробки, оновлення дисплея та подій контролера. З міркувань безпеки Marlin перезавантажиться, якщо центральний процесор занадто перевантажений для зчитування датчиків.

Marlin можна повністю керувати з хоста або в автономному режимі з SD-карти. Навіть без РК-контролера автономний друк у форматі SD все ще може бути ініційований із хоста, тому комп’ютер може бути відключений від принтера.

Програмне забезпечення для хостів доступне для декількох платформ, включаючи настільні системи, RaspberryPi та планшети Android. Будь-який пристрій з USB-портом та послідовним терміналом може технічно виступати в ролі хоста, але ви матимете кращий досвід друку за допомогою програмного забезпечення хоста, спеціально розробленого для 3D-принтерів. Поточний вибір включає:

- Pronterface– це хост з відкритим кодом від Kliment;
- RepetierHost– це хост із закритим кодом від RepetierSoftware;
- OctoPrint– це хост із відкритим кодом для RaspberryPi від GinaHäußge;
- Cura– це хост з відкритим кодом від Ultimaker;
- Simplify3D включає як хост, так і слайсер.

Багато 3D-принтерів постачаються з індивідуальною версією Repetier або Cura. Хоча це допомагає пов’язати бренд принтера із супутнім програмним забезпеченням, ці версії, як правило, застарілі та неоновлюються.

3.1.2 Klipper

Klipper - це спеціальна прошивка, призначена для обробки руху кінематики на RaspberryPi та спрощення роботи існуючої материнської плати 3D-принтера. Це забезпечує дуже точну синхронізацію крокового двигуна та потенційно набагато вищі швидкості руху.

Також Klipper працює з Octoprint. Це дозволяє керувати принтером за допомогою звичайного веб-браузера. Той самий RaspberryPi, який запускає Klipper, також може запускати Octoprint.

Основними особливостями Klipper є:

- високоточний кроковий рух. Klipper використовує повноцінний потужний одноплатний комп’ютер (наприклад, недорогий RaspberryPi) для розрахунку руху принтера. Він визначає, коли слід крокувати кожен кроковий

двигун, стискає ці події, передає їх на мікроконтролер, а потім мікроконтролер виконує кожну подію в заданий час. Кожна подія кроку запланована з точністю до 25 мікросекунд або вище. Програмне забезпечення не використовує кінематичні оцінки (наприклад, алгоритм Брезенхама) - натомість воно розраховує точні часи кроків на основі фізики прискорення та фізики кінематики машини. Точніший крековий рух забезпечує більш тиху та стабільну роботу принтера;

– висока швидкодія. Klipper здатний досягти високих швидкостей кроків як на нових, так і на старих мікроконтролерах. Навіть старі 8-бітні мікроконтролери можуть досягати швидкості понад 175 тисячкроків в секунду. На новіших мікроконтролерах можливі швидкості понад 500 тисячкроків в секунду. Вищі крокові швидкості забезпечують більшу швидкість друку. Час крокової події залишається точним навіть на високих швидкостях, що покращує загальну стабільність;

– підтримка принтерів з декількома мікроконтролерами. Наприклад, один мікроконтролер можна використовувати для управління екструдером, тоді як інший - нагрівачі принтера, а третій –для інших частин принтера. Програмне забезпечення Klipper реалізує синхронізацію годинника, щоб врахувати зсув годинника між мікроконтролерами. Для активації декількох мікроконтролерів не потрібен спеціальний код - для цього потрібно лише кілька додаткових рядків у файлі конфігурації;

– налаштування за допомогою простого конфігураційного файлу. Не потрібно перепрошивати мікроконтролер, щоб змінити налаштування. Вся конфігурація Klipper зберігається у стандартному конфігураційному файлі, який легко редагувати. Це полегшує налаштування та обслуговування обладнання;

– підтримка “SmoothPressureAdvance” - механізм, що враховує вплив тиску в екструдері. Це зменшує "сочання" екструдера та покращує якість кутів друку. Реалізація Klipper не передбачає миттєвих змін швидкості екструдера, що покращує загальну стабільність та міцність;

– Klipper підтримує “InputShaping”, щоб зменшити вплив вібрацій на якість друку. Це може зменшити або усунути "дзвін" (також відомий як "ореол",

"луна" або "брижі") на об’єктах, що друкуються. Це також може дозволити досягти більш високої швидкості друку, зберігаючи при цьому високу якість друку;

– використання “ітеративного розв’язувача” для обчислення точних часів кроків за простими кінематичними рівняннями. Це полегшує перенесення Klipper на нові типи принтерів, а також забезпечує точність хронометражу навіть при складній кінематиці (не потрібна “сегментація ліній”);

– Klipper працює на мікроконтролерах на базі ARM, AVR та PRU. Існуючі принтери у стилі “reprap” можуть запускати Klipper без будь-яких апаратних змін. Внутрішняорганізація сирцевого коду Klipper полегшує підтримку інших архітектур мікроконтролерів;

– спеціальні програмовані макроси. Нові команди G-Code можна визначити у файлі конфігурації принтера (зміни коду не потрібні). Ці команди можна програмувати - це дозволяє їм виконувати різні дії залежно від стану принтера;

– вбудований сервер API. На додаток до стандартного інтерфейсу G-Code, Klipper підтримує розширений інтерфейс програми на основі JSON. Це дозволяє програмістам створювати зовнішні програми з детальним управлінням принтером.

3.1.3 RepRap

RepRapFirmware працює на 32-розрядних мікропроцесорах, таких як процесори Atmel / Microchip SAM3X8E, SAM4E8E, SAM4S8C та SAME70Q20, знайдені в електроніці Duet. Існує також порт для плат, що використовують процесори LPC1768 / 1769, і працює процесор dpr STM STM. Він підтримує 3D-принтери, верстати з ЧПУ, лазерні різаки та гравери.

Це мікропрограмне забезпечення попередньо скомпільоване та прошито на платі принтера. Файли конфігурацій розташовані на SD-карті, підключеній до електроніки принтера. Таким чином, звичайним користувачам не потрібно компілювати програмне забезпечення та встановлювати будь-які засоби розробки.

Програмне забезпечення може отримувати G-код з USB-порту, послідовного порту, SD-карти, інтерфейсу Ethernet або WiFi через httpта інтерфейсу Ethernet через Telnet.

Хоча він може працювати від інтерфейсу, підключеного через USB, більшість користувачів воліє працювати через веб-інтерфейс, причому плата може бути підключена до мережі з допомогою портуEthernet або WiFi. Файли для друку завантажуються на SD-карту через веб-інтерфейс.

Філософія RepRapFirmware полягає в тому, що кожна операція виконується G-кодом, включаючи конфігурацію. З цієї причини RepRapFirmware підтримує ряд кодів GCodes, переважно в діапазонах M500-M599 та M900-M999, які в даний час не підтримуються більшістю інших прошивок.

Файл конфігурації зчитується на SD-карті під час запуску.

Будь-який G-код або макрос можуть бути надіслані на плату під час роботи з принтером, що забезпечує миттєвий зворотний зв’язок для будь-яких змін конфігурації. Оскільки інтерактивні модифікації втрачаються при наступному запуску плати, успішно протестовані G-коди вручну вводяться у файл конфігурації, який можна редагувати безпосередньо у веб-інтерфейсі.

Ця інтерактивна конфігурація робить введення та налаштування принтера простішою, ніж у більшості інших прошивок.

Автоматизація можлива за допомогою макросів, які можуть містити умовні інструкції та цикли.

3.1.4 Repetier

Repetier-Firmware - це прошивка для RepRap, як 3d-принтер, що працює на контролері, сумісному з arduino. Одною з основних особливостей даної прошивки є висока швидкість та надійність комунікації з принтером. Прошивка використовує контрольні суми для перевірки цілісності даних. Для цього потрібно програмне забезпечення хоста, яке надсилає команди, включаючи контрольну суму та номер рядка. Це справедливо для всього програмного забезпечення, сумісного з RepRap. Контрольна сума будується як просте 8-бітове значення xor

над командою. Це надає деякий простір для помилок, які неможливо виявити за допомогою мікропрограми, наприклад, двічі додавати додатковий символ, що не впливає на контрольну суму.

Для подолання цього недоліку в прошивку було додано новий протокол двійкових даних. Цей новий протокол Repetier використовує двійкову передачу даних та 16-бітову контрольну суму. Двійкові дані прискорюють аналіз команд і зменшують необхідні дані до приблизно 50% порівняно іхстандартною передачі ascii. 16-розрядна контрольна сума Флетчера робить спілкування дуже надійним. Наразі лише Repetier-Host підтримує надсилання даних у цьому форматі. Ви все ще можете використовувати інше хост-програмне забезпечення, наприклад Pronterface. Прошивка автоматично визначає, чи є команда у форматі ascii або двійковому форматі.

Для швидкого зв'язку може бути використаний широкий діапазон швидкості передачі даних. Для найбільш поширених плат на частоті 16 МГц рекомендується швидкість 250000бод, якщо використовуване програмне забезпечення хоста підтримує це налаштування.

3.1.5 Взаємодія з принтером

Зваживши на всі переваги та недоліки описаних прошивок було прийнято рішення використовувати саме Marlin, адже дане програмне забезпечення є найбільш розповсюдженим, підтримує широкий спектр конструкцій принтерів, активно розвивається та підтримує інтеграцію з системами-хостами.

Для реалізації системи моніторингу було вибрано хостOctoPrint через його популярність, відкритість та простоту. OctoPrint забезпечує веб-інтерфейс для управління 3D-принтерами, що дозволяє користувачеві розпочати завдання друку, надіславши G-код на 3D-принтер, підключений через USB. OctoPrint контролює стан завдання друку, а також самого принтера, в першу чергу температуру друкуючої головки (гарячий кінець) та температуру поверхні для друку, якщо поверхня для друкуможе нагріватись. OctoPrint також може відображати вихідні

дані підключеної веб-камери для відстеження стану друку, а також може візуалізувати G-код синхронізовано із завданням друку або асинхронно.

OctoPrint також надає систему плагінів, що дозволяє користувачам розширити функціональність. На даний момент в офіційному сховищі плагінів перелічено понад 150 плагінів.

OctoPrint може працювати на різних системах, але зазвичай працює на RaspberryPi. Дистрибутив під назвою OctoPi, заснований на ОС Raspbian для RaspberryPi, пропонує попередньо налаштовану версію OctoPrint, а також підтримку mjpeg-стримера для веб-камер.

Зазвичай користувачі, які запускають OctoPrint, відстежують прогрес друку через веб-камеру, приєднану до їх RaspberryPi. Однак, якщо веб-камери немає або потрібний інший спосіб перевірити друк, програма перегляду G-коду дозволяє це зробити.

3.2 Програмне забезпечення для роботи з зображеннями та 3D моделями

3.2.1 Обробка зображень та відео

В якості бібліотеки для роботи з зображеннями було обрано версію бібліотеки OpenCV для мови програмування Python. OpenCV – це бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), вистежування руху об'єктів, перетворення зображень, застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях.

Спочатку OpenCV був розроблений на C ++. На додаток до цього були надані прив'язки Python та Java. OpenCV працює на різних операційних системах, таких як Windows, Linux, OSx, FreeBSD, Net BSD, Open BSD тощо.

Нижче наведені основні бібліотечні модулі бібліотеки OpenCV.

Основна функціональність. Цей модуль охоплює основні структури даних, такі як Скаляр, Точка, Діапазон тощо, які використовуються для побудови додатків OpenCV. На додаток до них, він також включає багатовимірний масив Mat, який використовується для зберігання зображень.

Обробка зображень. Цей модуль охоплює різні операції з обробки зображень, такі як фільтрація зображень, геометричні перетворення зображень, перетворення кольорового простору, гістограми тощо.

Відео. Цей модуль охоплює такі концепції відео аналізу, як оцінка руху, віднімання фону та відстеження об’єктів.

Зчитування та генерація відео. Цей модуль реалізуєвідео зйомку та відеокодеки за допомогою бібліотеки OpenCV.

Calib3d. Цей модуль включає алгоритми, що стосуються основних алгоритмів геометрії декількох видів, калібрування одинарної та стерео камери, оцінки пози об'єкта, стерео відповідності та елементів 3D-реконструкції. [11]

Features2d. Цей модуль включає поняття виявлення та опису ознак.

Objdetect. Цей модуль включає виявлення об'єктів та екземплярів заздалегідь визначених класів, таких як обличчя, очі, гуртки, люди, машини тощо.

Highgui. Це простий у використанні інтерфейс з простими можливостями інтерфейсу.

3.2.2 Робота з 3D моделями

Для обробки та візуалізації 3D моделей було обрано бібліотекуOpenGL.OpenGL – відкрита графічна бібліотека, яка є одним з найпопулярніших прикладних програмних інтерфейсів для розробки додатків в області двовимірної і тривимірної графіки.

Бібліотека налічує близько 300 різних команд, які програміст використовує для завдання об'єктів і операцій, необхідних для написання інтерактивних графічних додатків. Використовується також при створенні комп'ютерних ігор, САПР, віртуальної реальності, візуалізації в наукових дослідженнях. [10]

Бібліотека OpenGL досить проста у використанні і навчанні, має дуже широким спектром можливостей. Ось деякі з її переваг:

– стабільність,OpenGLце стандарт. Всі зміни, що вносяться до нього, анонсуються заздалегідь і реалізуються так, щоб вже існуюче ПЗ не видавало помилок на нових графічних картах;

– надійність,всі функції, які залежать OpenGL, гарантують однаковий візуальний результат, незалежно від устаткування і операційної системи;

– переносимість, програми, що використовують OpenGL, можуть запускатися на різних архітектурах і під різними операційними системами (за умови перекомпіляції додатка, тобто OpenGL забезпечує переносимість на рівні вихідних кодів).

На сьогоднішній день графічна система OpenGL підтримується більшістю виробників апаратних і програмних платформ.

OpenGL не зберігає інформацію про "об'єкт". Все, що бачить OpenGL – це набір трикутників та контекст, за допомогою якого їх можна зобразити.

Через це загальним способом використання OpenGL є намалювати все, що потрібно для малювання, а потім показати це зображення за допомогою команди заміни буфера, що залежить від платформи. За потреби оновити зображення, необхідно виконати візуалізацію усього кадру, навіть якщо потрібно оновити лише частину зображення.

3.3 Програмні засоби для роботи з нейронними мережами

Для того, щоб реалізувати виявлення неполадок друку з допомогою згорткових нейронних мереж було використано бібліотеку Tensorflow та її модуль Keras. TensorFlow - це наскрізна відкрита платформа для машинного навчання. Він має всеосяжну, гнучку екосистему інструментів, бібліотек та ресурсів спільноти, що дозволяє дослідникам використовувати найсучасніші технології машинного навчання, а розробникам легко створювати та розгортати додатки на його базі.

Спочатку TensorFlow був розроблений дослідниками та інженерами, що працюють у команді GoogleBrain в рамках організації GoogleIntelligenceResearchMachine для проведення машинного навчання та дослідження глибоких нейронних мереж. Однак, система є достатньо загальною, щоб бути застосовною і в багатьох інших доменах.

TensorFlow надає стабільні API-інтерфейси для мов програмування Python та C ++, а також гарантований зворотньо сумісний API для інших мов.

API TensorFlowє ієрархічними, причому API високого рівня побудовані на API низького рівня. Дослідники машинного навчання використовують API низького рівня для створення та вивчення нових алгоритмів машинного навчання.

Існує три основні конструкції для операцій TensorFlow: вектори, масиви (матриці), тензори. Вектор - це математичний об’єкт, який має напрямок і величину. За його допомогою знаходять положення однієї точки в просторі щодо іншої точки. Масив - це розташування або ряд елементів, таких як символи, цифри або вирази. Масиви можуть бути n-мірними, тому матриця - це масив із 2 вимірами. Тензор - це об’єкт, що описує лінійний зв’язок між скалярами, векторами та іншими тензорами. Іншими словами, це укладання декількох масивів для створення вищих розмірних структур. Практичним прикладом тензора в дії є зображення. Коли обробляється зображення RGB, це тривимірний тензор із шарами для кожного кольору в розмірах висоти та ширини.[12]

Для представлення графа обчислень в TensorFlowвикористовується спеціальна структура графа. Графи в TensorFlow - це структури даних, що містять набір об'єктів tf.Operation. Графи корисні, оскільки їх можна зберігати, запускати та відновлювати без оригінального коду Python. Граф TensorFlow - це певний тип спрямованого графа, який використовується для визначення обчислювальної структури. У графаTensorFlow є три основні об'єкти, які можна використовувати для операцій. По-перше, це константа. Вона створює вузол, який приймає значення і не змінюється. Наступним об'єктом графа в TensorFlow є змінна. Змінні - це вузли, що містять стан, це означає, що вони можуть зберігати своє значення при багаторазовому виконанні графа.. Кінцевим об’єктом є заповнювач.

Заповнювачі замінюються на конкретні значення під час виконання на графу, тобто можуть відрізнятись між запусками. Вони використовуються, коли граф залежить від зовнішніх даних. Для запуска графу необхідний об’єкт сесії, який зберігає метаінформацію про конкретний запуск графу та значення заповнювачів.

Для того щоб спростити процес побудови та навчання нейроннох мережі було використано бібліотеку Keras. Keras – це високорівневий API Tensorflow, високопродуктивний інтерфейс для вирішення проблем машинного навчання з акцентом на сучасне глибоке навчання. Він надає основні абстракції та будівельні блоки для розробки та транспортування рішень для машинного навчання з високою швидкістю ітерацій розробки.

Keras надає інженерам і дослідникам можливість повною мірою скористатися масштабованістю та крос-платформними можливостями TensorFlow.Keras можливо запустити на TPU або на великих кластерах графічних процесорів, а також експортувати моделі Keras для запуску в браузері або на мобільному пристрої.

Keras містить численні реалізації загальновживаних нейромережевих будівельних блоків, таких як шари, функції активації, оптимізатори та безліч інструментів для спрощення роботи з даними, зображеннями та текстом для спрощення написання та навчання нейронних мереж.

На додаток до стандартних нейронних мереж, Keras підтримує згорткові та рекуренті нейронні мережі, а такожнадає інші загальноприйняті утиліти, такі як пакетна нормалізація та семплування даних.

3.4 Програмні засоби для потокової передачі відео

Для реалізації потокової передачі відео було вирішено використовувати протокол MJPEG. MJPEG (Motion JPEG) - покадровий метод відеостискання, основною особливістю якого є стиснення кожного окремого кадру відеопотоку за допомогою алгоритму стиснення зображень JPEG.

При стисненні методом MJPG міжкадрова різниця не враховується.

MJPEG широко застосовується в наступних областях:

- веб-камери;
- нелинійнийвідеомонтаж;
- IP-камери;
- системи відеоспостереження;
- цифрові фотоаппарати.

M-JPEG має вбудовану підтримку QuickTimePlayer, консолі PlayStation та браузерів Safari, GoogleChrome та MozillaFirefox.

Motion JPEG використовує покадрове стискання із втратами на основі дискретного косинусного перетворення (ДКП). Ця математична операція перетворює кожен кадр або частину зображення із просторової області в частотну. Психовізуальна модель, заснована на особливостях відтворення зображень людиною, використовує зазвичай грубе квантування високочастотної складової зображення та більш акуратне квантування низькочастотної складової, знижуючи тим самим точність передачі різких переходів яскравості та відтінків кольорів. Квантовані коефіцієнти ДКП без втрат упаковуються у вихідний бітовий потік із використанням кодів Хаффмана або за допомогою арифметичного кодування. Практично усі програмні реалізації MJPEG дозволяють користувачам контролювати степінь стискання (а також інші параметри) та досягати компромісної якості зображення та розміру файлу. При апаратних рішеннях параметрів кодування, як правило, попередньо вибрані та зафіксовані.

Заголовок кожного кодованого MJPEG зазвичай відповідає стандарту JPEG, однак, допустимими є деякі невідповідні стандарти. Так, наприклад, у ньому може бути відсутнім маркер DHT, що визначає таблиці для хаффмановського декодування. У цьому випадку при обробці декодувань слід використовувати таблиці, включені в розділ K.3 стандарту JPEG (CCITT Rec. T.81).

У MJPEG застосовується схема виключно внутрішньокадровогоскискання, на відміну від більш складного міжкадрового стискання. У цей час, як сучасні відеоформати з міжкадровим зв’язком, такі як MPEG1, MPEG2, H.264 / MPEG-4

AVC та їх подібні, досягають у середньому ступені стискання 1:50 і більше, відсутність у MJPEG міжкадровогозтискання, як правило, не дозволяє отримувати коефіцієнт зжатості, що перевищує в 1:20, залежно від допустимості просторових пошуків у декодованих кадрах відеопослідовності. Так як кадри зберігають незалежність одного від іншого, MJPEG вимагає менших обчислювальних ресурсів та оперативної пам'яті на етапі кодування. Однак, декодування MJPEG може стати більш затяжним, що при використанні міжкадрового стискання, внаслідок, першочергового, передбачається повне декодування в MJPEG кожного зображення макроблока, тоді як при використанні схеми з міжкадровимзтискання частини макроблоків, розміщені як «пропустити», не декодується, а береться з попередніх кадрів. По друге, час виконання процедур хафменовського декодування та зворотнього ДКП залежить від інформаційної насиченості декодованого макроблока зображення, яке внаслідок відсутності міжкадрового зв’язку виявляється значно більшим, ніж за його наявності (в першому випадку декодується повне зображення, в другому - різниця).

У межах внутрішньокадрового стискання зображення в MJPEG якість зображення залежить безпосередньо від статичної (просторової) складності кожного відеокадра. Кадри з більшими гладкими переходами або монотонними областями добре зберігаються, але при занадто високих рівнях змісту вмісту, окрім оригінальних деталей, видно артефакти змісту у відео блоках розміром 8x8 пікселів, невелика різниться по яскравості та відтінку кольорів. Поява їх пов'язано з грубим квантуванням низькочастотних коефіцієнтів ДКП. Кадри, що надають складні текстури, тонкі криві лінії, окрім артефактів блочності містять також артефакти, що проявляються у вигляді шуми навколо тонких ліній і на різких границях (так званий ефект Гіббса), пов'язані з групами квантування високоякісних коефіцієнтів ДКП. [13]

Для форматів QuickTimeApple визначила два типи кодування: MJPEG-A та MJPEG-B. MJPEG-B не зберігає структуру файлів JPEG у відеофайлі, відповідно, неможливо вивести кадр у файл JPEG без реконструкції заголовка JPEG.

Основним перевагою відеозв'язкуMotion JPEG є простота реалізації, яка робить MJPEG підходящим для реалізації на пристроях з обмеженими обчислювальними ресурсами.

Надзвичайно швидкий нелінійний відеомонтах – якщо який-небудь кадр бере участь в цілому (без змін) з одного MJPEG-джерела, його можна записати у вихідний MJPEG-поток як є, без декодування-стискання.

За високого бітрейті MJPEG дає якісні стоп-кадри, що дозволяє використовувати його, наприклад, у системах відеоспостереження (тоді це потрібно, наприклад, для виявлення номера проїхавшого автомобіля). Однак при відсутності мідкадрового стискання досягнення заданого бітрейту вимагає використання більшого стискання, що призводить до появи артефактів.

Недоліками MJPEG є більш низький коефіцієнтом зтисканння у порівняні з потоковими методами передачі відео (наприклад, MPEG-4) та поява артефактів стискання при високих коефіцієнтах стискання.

З огляну на існуючі недоліки та переваги протокол MJPEGідеально підходить для передачі відео для виявлення неполадок у процесі друку, оскільки кодування відео відбуватиметься на пристрої з достатньо обмеженими ресурсами, а також потребуватиме покадрового аналізу, який працює дуже добре при використанні даного підходу.

3.5 Архітектура програмного забезпечення

Зважаючи на описані вище технології, та необхідність взаємодії з 3D принтером для написання програмного забезпечення було обрано клієнт-серверну архітектуру, що складається з кількох компонент.

Для написання програмного забезпечення використовувалась мова програмування Python, оскільки для неї є всі необхідні клієнтські бібліотеки для взаємодії з зазначеними технологіями.

Діаграма розгортання наведена на рисунку 3.1.

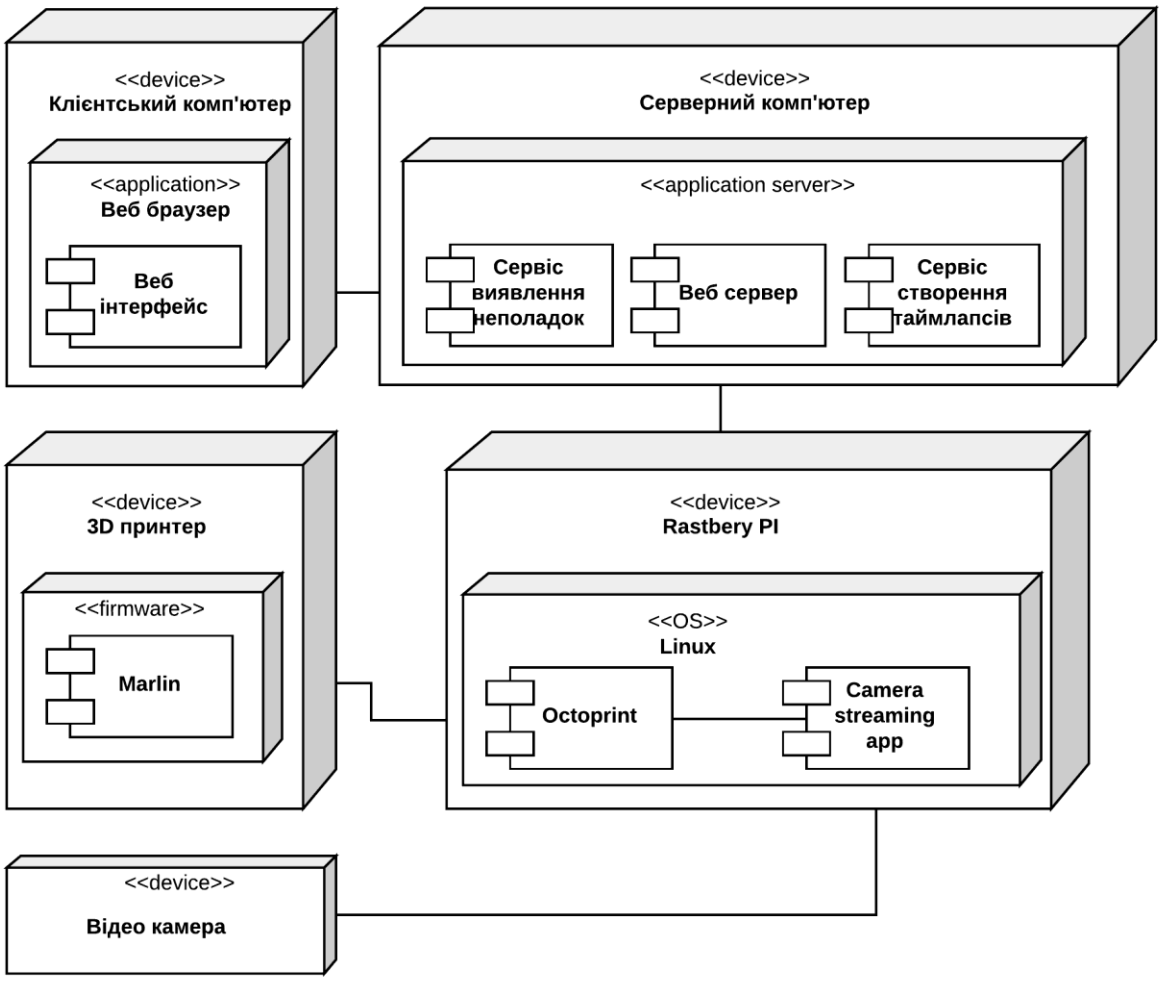


Рисунок 3.1 – Діаграма розгортання системи

Як зазначено на діаграмі система складається з кількох компонентів. Для взаємодії з 3D принтером до нього підключений односхемнийкопм’ютерRaspberryPi, на якому виконується Octoprint та програмне забезпечення для передачі потокового відео з відео камери, яка, як і принтер підключена до RaspberryPi.

Для зменшення навантаження на даний комп’ютер та спрощення підключення до системи основні обчислення відбуваються на окремому сервері, дані до якого поступають з Octoprint. Саме на даному сервері розгорнутий веб сервер та відбувається процес аналізу відео друку та створення таймлапсів.

Щоб реалізувати доступ до сервісу через веб інтерфейс було вибрано фреймворк для написання веб застосунків Flask. Flask вважається більш пітонічним, ніж веб-фреймворкDjango, оскільки в загальних ситуаціях

еквівалентний веб-додаток Flask є більш явним. З Flask також легко розпочати роботу початківцям, оскільки для створення і запуску простого додатка необхідна дуже мала кількість коду, який просто зрозуміти. [14]

Flask має широкий спектр доступних розширень, що дозволяє інтегрувати власні рішення щодо зберігання, взаємодії з базами даних, автентифікації та авторизації, безпеки тощо. На інтеграцію та налаштування вашого додатку знадобиться час, але додатки можна створювати поступово і не включатимуть бібліотеки та код для речей, які ваша програма не використовує.

Для взаємодії між компонентами системи було обрано підхід REST. Дизайн REST або RESTful API (Репрезентативна передача стану) призначений для використання переваг існуючих протоколів. Хоча REST можна використовувати майже за будь-яким протоколом, він зазвичай використовує переваги HTTP, коли використовується для веб-API. Це означає, що розробникам не потрібно встановлювати бібліотеки або додаткове програмне забезпечення, щоб скористатися перевагами дизайну REST API. Дизайн REST API був визначений доктором Роєм Філдінгом у його докторській дисертації 2000 року. Він примітний своєю неймовірною гнучкістю. Оскільки дані не прив'язані до методів та ресурсів, REST має можливість обробляти кілька типів викликів, повертати різні формати даних і навіть структурно змінюватись за допомогою правильної реалізації гіпермедіа.

Ця свобода та гнучкість, властиві дизайну REST API, дозволяють вам створити API, який відповідає вашим потребам, одночасно задовольняючи потреби дуже різноманітних клієнтів. На відміну від SOAP, REST не обмежується XML, а натомість може повертати XML, JSON, YAML або будь-який інший формат, залежно від запитів клієнта. І на відміну від RPC, користувачі не повинні знати імена процедур або конкретні параметри в певному порядку.

Однак у дизайну REST API є недоліки. Ви можете втратити здатність підтримувати стан у REST, наприклад, протягом сесій, а новішим розробникам може бути складніше використовувати. Важливо також зрозуміти, що робить REST API RESTful, і чому ці обмеження існують, перш ніж створювати ваш API.

Зрештою, якщо ви не розумієте, чому щось сконструйовано таким, яким воно є, ви можете перешкодити вашим зусиллям, навіть не усвідомлюючи цього.[15]

Веб інтерфейс розробленої системи зображено на рисунку 3.2.

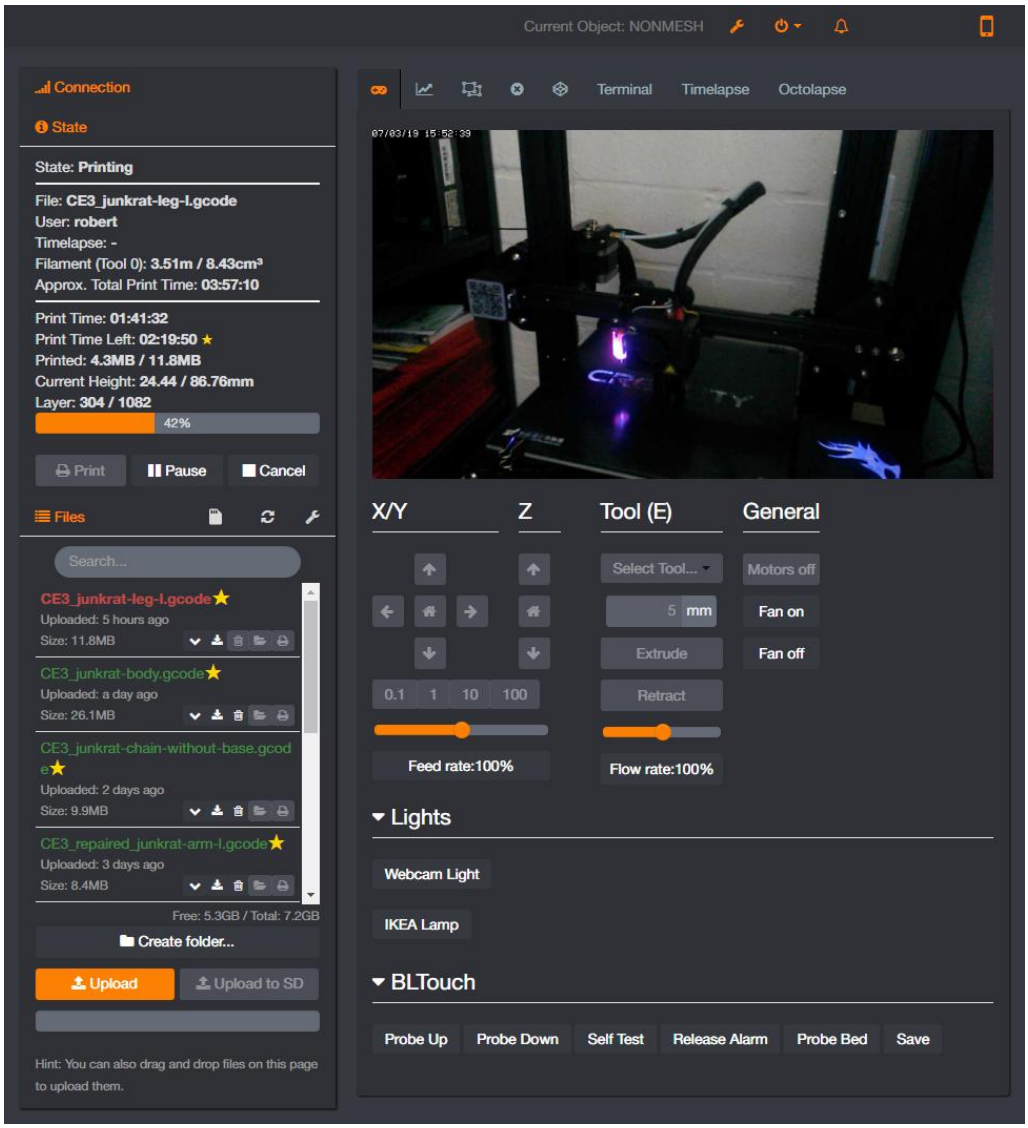


Рисунок 3.2 – Веб інтерфейс системи

Висновок розділу

У даному розділі було розглянуто технології, що використовувались при побудові програмного забезпечення, обгрунтованоїх вибір, після чого наведено загальну архітектуру системи, схему її розгортання та описано взаємодію компонентів між собою.

В якості прошивки принтера було обрано Marlin, що є популярним вибором для таких задач. Для взаємодії з прошивкою принтера використовується

Octoprint, що надає зручний інтерфейс для віддаленого управління принтером та легко розширюється з допомогою системи плагінів.

Для реалізації розпізнавання неполадок з допомогою комп’ютерного зору було обрано технології OpenCV та OpenGL, а для підходу з використанням нейронних мереж було застосовано фреймворкTensorflow та бібліотеку Keras.

Основною мовою програмування, що застосована у написання програмного забезпечення є Python. Таке рішення було прийнято через наявність клієнтських бібліотек для всіх необхідних технологій. Програмне забезпечення побудоване з кількох модулів, кожен з яких виконує специфічну функцію. Частина модулів виконуються на RasberryPi, що підключене до принтеру та камери. Вони передають всі необхідні дані на серверний застосунок, який їх обробляє, передає користувачу, а також віддає команди управління принтером.

4 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ

4.1 Ефективність методу на основі комп’ютерного зору

Для дослідження ефективності методу, що визначає наявність неполадки з допомогою комп’ютерного зору та аналізу 3D моделі об’єкту даний метод був застосований на тестовому об’єкті для друку, шо має форму літер “DTU”. У даному об’єкті літеру “T” неможливо успішно роздрукувати без підтримок з використанням технології FDM, оскільки об’єкт містить елементи, які повинні друкуватись у повітрі, не опираючись ні на що. Генерацію підтримок було навмисно вимкнено для демонстрації невдалого друку.

На рисунку 4.1 зображено результат сегментації від фону реального об’єкта зліва та його 3D моделі під час друкусправа.

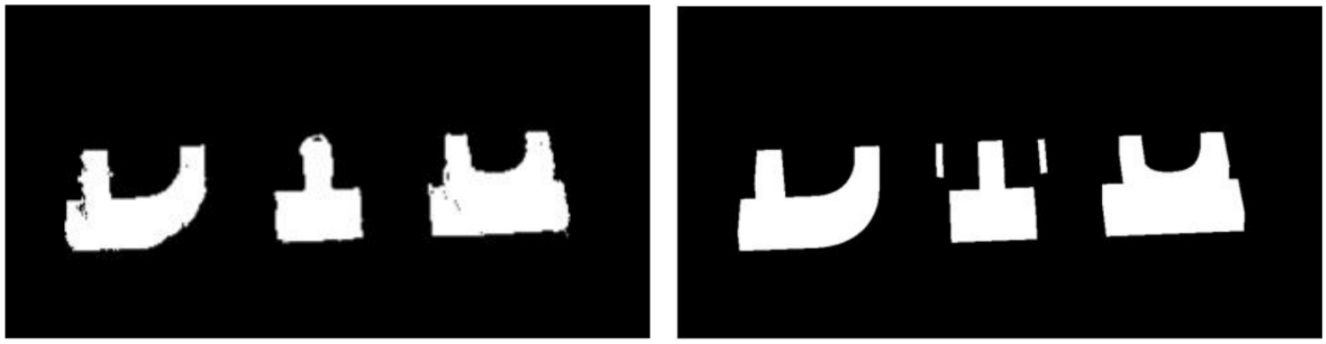


Рисунок 4.1 – Результат сегментації тестового об’єкта

Як і очікувалось, на реальному об’єкті відсутні верхні елементи літери “T”. Графік значення середньоквадратичного відхилення генерованого зображення та реального фото наведено на рисунку 4.2.

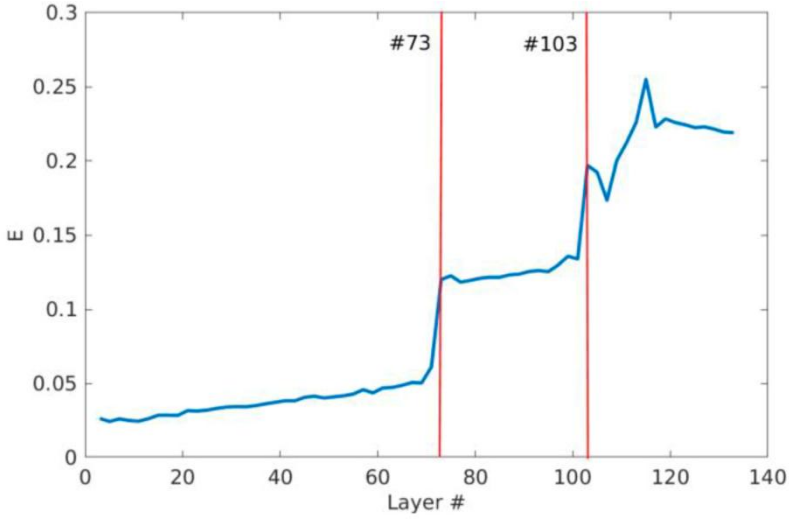


Рисунок 4.2 – Графік значення середньоквадратичного відхилення

По графіку чітко видно зростання значення відхилення на шарах номер 73 та 103, що свідчать про проблеми з друком. На рисунку 4.3 зображено стан об’єкта до виявлення неполадки зліва на шарі 73 та після цього справа. Для наглядної демонстрації сегментоване зображення було накладено як фільтр на реальне зображення з камери і пікселі, що не співпадають між реальним та генерованим зображенням зафарбовані червоним кольором.

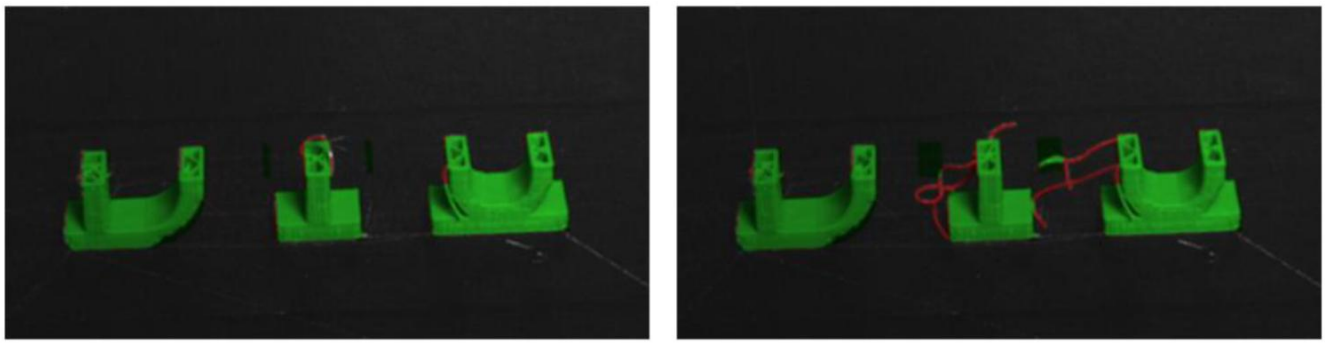


Рисунок 4.3 – Виявлення неполадки на шарі з номером 73

Ситуація, яка виникла на шарі під номером 103 показана на рисунку 4.4.

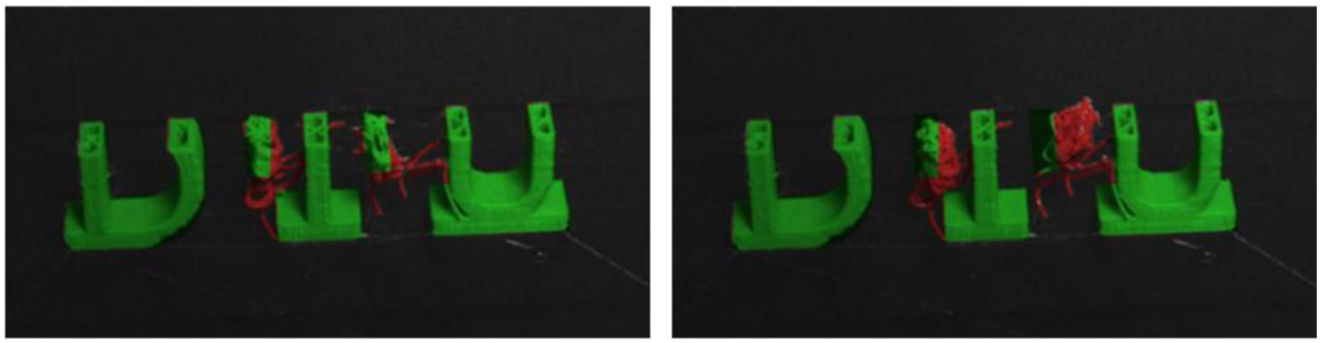


Рисунок 4.4 – Виявлення неполадки на шарі з номером 104

Таким чином, можна впевнено говорити про високу ефективність розробленого методу. Схожу ситуацію можна спостерігати і при інших типах неполадок, що детально описані у розділі 2.

4.2 Ефективність методу на основі згорткових нейронних мереж

Як було сказано при описі даного методу, для навчання нейронної мережі було застосовано штучно генерований набір даних. Для цього були штучно

згенеровано випадкові тривимірніклубки пластику. Після чого на отримані 3D моделі накладені тіні від штучних джерел світла та згенеровані зображення накладені на кадри з реальним процесом друку. Приклад такого згенерваного клубка наведений на рисунку 4.5.

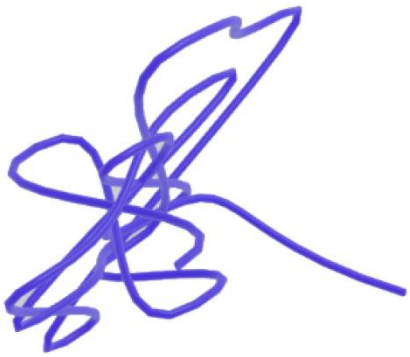


Рисунок 4.5 – Штучно згенерований клубок пластику

Після накладання на відео з реальним процесом друку було отримано результат, зображений на рисунку 4.6.

Рисунок 4.6 – Штучно згенеровані дані для навчання нейронної мережі

Після генерації даних для навчання було проведено навчання мережі. Графік точності виявлення проблеми для тестової та валідаційної вибірки на згенерованих даних наведено на рисунку 4.7.

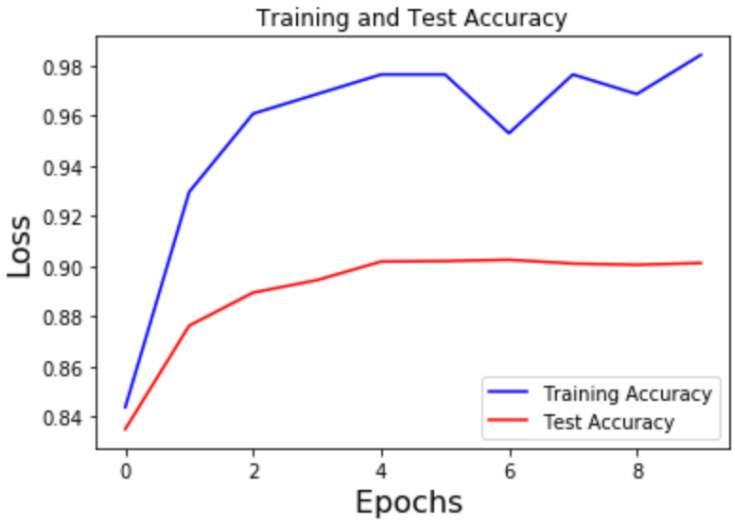


Рисунок 4.7– Точність роботи нейронної мережі

Як видно з графіку, навіть на тестовій вибірці створеній з згенерваних зображень точність не перевищує 90%, чого не достатньо для надійного моніторингу, оскільки є висока вірогідність false-positive результатів, чого не можна допускати у подібних системах моніторингу, оскільки вірогідність помилкової зупинки друку занадто висока. На реальних, не згенерованих даних результати ще гірші і не перевищують точність 70%.

Для покращення якості розпізнавання необхідно зібрати великий набір реальних даних, що на сьогоднішній день є непосильною задачею.

Висновок розділу

В даному розділі було досліджено ефективність роботи представлених у попередніх розділі методів. Метод на основі комп’ютерного зору та аналізу 3D моделі показав високу точність виявлення неполадок, надійність та простоту у використанні. На противагу, метод, що базується на виявленні клубків пластику з використанням згорткових нейронних мереж на реальних даних показав точність, що не перевищує 70%, а на тестовій вибірці – 90%, що не дозволяє його використовувати як основний. Попри це, комбінація цих методів в загальному може покращити точність усієї системи в цілому.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

5.1 Опис ідеї проекту

Ідея проекту полягає у створенні програмного продукту, що дозволятиме проводити моніторинг процесу 3D друку та віддалено віддавати команди принтеру, в тому числі автоматична зупинка процесу друку у разі виявлення проблеми.

Детальний опис ідеї наведено в таблиці 5.1.

Таблиця 5.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробка програмного продукту, що дозволятиме проводити моніторинг процесу 3D друку та віддалено віддавати команди принтеру, в тому числі автоматична зупинка процесу друку у разі виявлення проблеми.	1. Виявлення неполадок в процесі 3D друку.	При використанні програмного продукту відпадає необхідність у постійному особистому контролі друку зі сторони оператора принтера.
	2. Перегляд стану друку у реальному часі.	З’являється можливість переглядати стан друку будь-де.
	3. Віддалене управління 3D принтером.	За потреби перервати процес друку користувач може це зробити віддалено.
	4. Автоматизація виробництва.	Систему можна використовувати для автоматизації процесу комерційного 3D друку.

На даний момент на ринку існує лише один конкурент даного продукту – програмний комплекс TheSpaghettiDetective. TheSpaghettiDetective надає дуже схожий функціонал, одна принцип роботи у мого продукту дещо відрізняється та вигідно виділяє його на фоні конкурента. Детальне порівняння наведено у таблиці 5.2.

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	Концепції конкурентів		Слабкі (W),нейтральні (N) тасильні (S) сторони		
		Мій проект	TheSpaghettiDetective	W	N	S
1.	Виявлення відділення об’єкта від поверхні друку	Так	Упосередковано, через виявлення клубків пластику, що іноді не надійно			+
2.	Виявлення зсуву шарів	Так	Упосередковано, через виявлення клубків пластику, що іноді не надійно			+
3.	Виявлення зупинки подачі матеріалу	Так	Ні			+
4.	Виявлення клубків пластику	Так	Так		+	
5.	Віддалене управління принтером	Так	Так		+	
6.	Перегляд відео друку у реальному	Так	Так		+	

	часі					

Продовження таблиці 5.2

№ п/п	Техніко-економічні характеристики ідеї	Концепції конкурентів		Слабкі (W), нейтральні (N) та сильні (S) сторони		
		Мій проект	TheSpaghettiDetective	W	N	S
7.	Створення та збереження таймлапсів	Так	Так		+	
8.	Урахування особливостей моделі при моніторингу	Так	Ні			+
9.	Можливість редагування функціоналу та розширення функціоналу	Так	Так		+	

5.2 Технологічний аудит ідеї проекту

Технологічну здійсненність ідеї проекту наведено у таблиці 5.3.

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технологія реалізації	Наявність технології	Доступність технології
1.	Виявлення відділення об’єкта від поверхні друку, зупинки подачі матеріалу, зсуву	OpenCV, OpenGL	Наявна	Безкоштовні, з відкритим сирцевим

	шарів, урахування особливостей моделі при моніторингу			кодом

Продовження таблиці 5.3

№ п/п	Ідея проекту	Технологія реалізації	Наявність технології	Доступність технології
2.	Виявлення клубків пластику	OpenCV, Tensorflow, Keras, scikitlearn	Наявна	Безкоштовні, з відкритим сирцевим кодом
3.	Віддалене управління принтером	OctoPrint	Наявна	Безкоштовні, з відкритим сирцевим кодом
4.	Перегляд відео друку у реальному часі	Стрімінгова передача відео	Наявна	Безкоштовні та платні
5.	Створення та збереження таймлапсів	Бібліотеки для створення та редагування відео	Наявна	Безкоштовні, з відкритим сирцевим кодом

Як видно з таблиці вище, всі необхідні для реалізації технології наявні на даний момент, а також є безкоштовними для комерційного використання. Таким чином проект є повністю можливим для реалізації.

5.3 Аналіз ринкових можливостей запуску стартап-проекту

У даному пункті буде розглянуто ринкові можливості стартап-проекту, а також необхідні для виводу на ринок умови, потенційно загрози та можливості, що впливатимуть на розвиток проекту.

Аналіз поточної ситуації на ринку наведено у таблиці 5.4.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекті

№ п/п	Показники стану ринку (найменування)	Характеристика
1.	Кількість головних гравців, од	1
2.	Загальний обсяг продаж, грн/міс	1400 грн/міс
3.	Динаміка ринку	Зростає
4.	Наявність обмежень для входу	Немає
5.	Специфічні вимоги до стандартизації та сертифікації	Немає
6.	Середня норма рентабельності в галузі, %	Дані відсутні

Судячи з даних наведених вище ринок є достатньо привабливим через наявність лише одного конкурента, стрімкий ріст та відсутність обмежень для входу.

У таблиці 5.5 наведені групи потенційних клієнтів проекту, їх характеристики та вимоги.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Можливість автономного моніторингу процесу 3D друку та віддаленого управління та спостереженням за 3D принтером	Власники домашніх 3D принтерів, мейкери, власники бізнесу малого та середнього, що базується на виготовленні об’єктів з допомогою 3D друку	Кількість принтерів, конструкція принтерів, Прошивка принтерів, знання у налаштуванні програмного забезпечення	Наявність веб інтерфейсу, мобільного додатку, зручність та простота у використанні
2.	Автоматизація 3D друку	Власники бізнесу малого та середнього, що базується на виготовленні об’єктів з допомогою 3D друку	Кількість принтерів, конструкція принтерів, прошивка, знання комп’ютерних систем	Наявність REST API, якісна документація, відкритий сирцевий код, модульність

Фактори потенційних загроз для стартап-проекту наведено у таблиці 5.6.

Таблиця 5.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Конкуренція	Поява нових конкурентів на ринку	1) Зміна цінової політики; 2) запозичення функціоналу у конкурентів; 3) розширення рекламної компанії.
2.	Актуальність технологій	Зменшення популярності програмного забезпечення для управління принтером, що використовується як компонент системи	1) Розширення спектру прошивок принтерів, що підтримуються; 2) Реалізація системи плагінів, що дозволить користувачам без зусиль додавати бажаний функціонал.
3.	Непрогнозована велика популярність системи	Через непрогнозоовано великий попит на систему невистачає	1) Перехід на хмарний хостинг з підтримкою динамчного горизонтального

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
		обчислювальних можливостей для її нормальної роботи	масштабування; 2) Збільшення вартості експлуатації системи.

Фактори потенційних можливостей наведені у таблиці 5.7.

Таблиця 5.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Зниження довіри до конкурента	Через ненадійність ПЗ конкурента його клієнтська база зменшується	1) У рекламній компанії акцентувати увагу на переваги продукту над конкурентом
2.	Зростання цільової аудиторії продукту	Через ріст популярності 3D друку цільова аудиторія сильно зросла	1) Колаборація з виробниками 3D принтерів для взаємної реклами та інтеграції системи в принтери “з заводу”

Ступеневий аналіз конкуренції на ринку представлено у таблиці 5.8.

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)

1. Вказати тип конкуренції - чиста	На ринку зараз існує лише одне рішення	Аналіз цінової політики конкурента та встановлення аналогічних цін при кращій якості послуг, що надаються, акцент над якісними перевагами над конкурентом у рекламній компанії

Продовження таблиці 5.8

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
2. За рівнем конкурентної боротьби - міжнародний	Товар та компанія ніяк не є прив'язаними до країни з точки зору кінцевого користувача	Локалізація інтерфейсу для країн з найбільшим попитом
3. За галузевою ознакою - внутрішньогалузева	Конкурент надає аналогічні послуги в середині даної галузі	Активна робота по збільшенню популярності галузі та співпраця з виробниками 3D принтерів для інтеграції системи у принтери одразу на виробництві
4. Конкуренція за видами товарів: - товарно-видова	Види товарів є однакові, відрізняється лише виконання	Якісні переваги перед програмним продуктом конкурентів за рахунок більш передового технологічного підходу

5. За характером конкурентних переваг - нецінова	Проект надає кращу якість моніторингу та більше можливостей ніж наявний конкурент	Якісні переваги перед програмним продуктом конкурентів за рахунок більш передового технологічного підходу
6. За інтенсивністю - марочна	Бренди присутні	Активна рекламна компанія з акцентом на перевагах порівняно з наявним аналогом

У таблиці 5.9 наведено аналіз конкуренції в галузі за М. Портером

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	TheSpaghettiDetective	Немає	Постачальники відсутні	Власники домашніх 3D принтерів та власники бізнесу, що базується на 3D друці	Товари замінники відсутні
Висновки:	Існує всього один конкурент, що технологічно програє даному	Є можливість виходу на ринок	Постачальники не диктують умови,	Для користувача важлива висока	Оскільки товари замінники

	продукту та вже давно не розвивається	у термін близько одного місяця	оскільки даний продукт є самостійним	надійність моніторингу та зручність використання веб інтерфейсу та мобільного додатку	відсутні, обмежень для роботи на ринку немає

За результатами аналізу конкуренції можна сказати, що ринок є відкритим для входу та потребує товару, що буде якісно кращим за наявне рішення монополіст.

Обґрунтування факторів конкурентоспроможності наведено у таблиці 5.10.

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1.	Якість моніторингу	Єдине наявне рішення конкурент є неефективним та не може бути застосоване для більшості реальних завдань.
2.	Зручність інтерфейсу	Наявний конкурент не надає інтуїтивного інтерфейсу та не має мобільного додатку, що відштовхує багатьох користувачів від його використання
3.	Відкритість системи	Оскільки специфіка галузі, у якій продукт застосовується тісно

		переплітається з комп'ютерним програмуванням, користувачам важливо щоб продукт не був закритим, адже це може суттєво звузити сферу його застосування

Порівняльний аналіз сильних та слабких сторін проекту наведено у таблиці 5.11.

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з проектом						
			-3	-2	-1	0	1	2	3
1.	Якість моніторингу	20		+					
2.	Зручність інтерфейсу	20		+					
3.	Відкритість системи	15				+			

У таблиці 5.12 наведено SWOT-аналіз стартап-проекту.

Таблиця 5.12 – SWOT- аналіз стартап-проекту

Сильні сторони: якість моніторингу, веб версія та мобільний додаток, зручність користування, відкритість системи	Слабкі сторони: бренд невідомий
Можливості: через ріст популярності 3D друку цільова аудиторія сильно	Загрози: конкуренція, актуальність технологій, важко прогнозований

зросла, конкурент давно не розвивається	попит.

Альтернативи ринкового впровадження стартап-проекту описані в таблиці 5.13.

Таблиця 5.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Підтримка альтернативних прошивок 3D принтерів	70%	1 місяць

Продовження таблиці 5.13

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
2.	Реалізація системи плагінів для розширення функціоналу кінцевими користувачами	50%	1.5 місяці

Проаналізувавши альтернативи було обрано реалізація підтримки альтернативних прошивок 3D принтерів, так як вона з більшою вірогідністю принесе результат та для її виконання необхідно менше часу.

5.4 Розробка ринкової стратегії проекту

У таблиці 5.14 наведено вибір цільових груп потенційних споживачів.

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних	Готовність споживачів сприйняти	Орієнтовний попит в межах	Інтенсивність конкуренції в сегменті	Простота входу у сегмент

	клієнтів	продукт	цільової групи (сегменту)		
1.	Власники домашніх 3D принтерів без глибоких технічних знань	Висока готовність	Високий попит	Відсутня, оскільки єдиним конкурентом практично не користуються	Низька складність

Продовження таблиці 5.14

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
2.	Власники малого та середнього бізнесу, що базується на 3D друці	Середня готовність	Високий попит	Відсутня, оскільки єдиним конкурентом не користуються	Середня складність
3.	Ентузіасти 3D друку, що розвивають цю технологію	Висока готовність	Дуже високий попит	Середня, оскільки серед цієї групи конкурент достатньо	Низька складність

				популярний	
Які цільові групи обрано: вирішено охопити всіх користувачів 3D принтерів, так як цей сегмент ще достатньо невеликий, хоча і стрімко розвивається, тому важливо максимально покрити потреби кожного підсегменту не зважаючи на відмінності між ними.					

Інформацію по визначенню базової стратегії розвитку наведено у таблиці 5.15.

Таблиця 5.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1.	Підтримка альтернативних прошивок 3D принтерів	Повне охоплення ринку	Якість моніторингу, зручність інтерфейсу, відкритість продукту, наявність веб та мобільної версії	Стратегія диференціації

У таблиці 5.16 описано визначення базової стратегії конкурентної поведінки.

Таблиця 5.16 – Визначення базової стратегії конкурентної поведінки

№	Чи є проект	Чи буде	Чи буде	Стратегія

п/п	«першопрохідцем» на ринку?	компанія шукати нових споживачів, або забирати існуючих у конкурентів?	компанія копіювати основні характеристики товару конкурента, і які?	конкурентної поведінки*
1.	Ні	Забирати існуючих та залучати нових	Так, модель монетизації та перелік основних можливостей	Стратегія виклику лідера

Опис визначення стратегії позиціонування наведено у таблиці 5.17.

Таблиця 5.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторі	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1.	Якість моніторингу, зручність інтерфейсу, відкритість	Стратегія диференціації	Якість моніторингу, зручність інтерфейсу, відкритість продукту, наявність веб та мобільної версії	Надійність, зручність, гнучкість

	продукту, наявність веб та мобільної версії			

5.5 Розроблення маркетингової програми стартап-проекту

У таблиці 5.18 описано визначення ключових переваг концепції потенційного товару.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключовіперевагиередконкурентами
1.	Якісний моніторинг	Моніторинг відбувається з високою надійністю, відловлюючи широкий спектр різноманітних неполадок та не даючи false-positive результатів	Висока надійність моніторингу та широкий спектр різноманітних неполадок з відсутністю false-positive результатів
2.	Зручний	Зручність у	Зручніший інтерфейс та наявність

	інтерфейс	користуванні як веб версією, так і мобільним додатком	мобільного додатку
3.	Автоматизація процесу друку	Можливість повністю автоматизувати процес 3D друку	Наявність можливості автоматизації
4.	Підтримка різних виробників 3D принтерів	Можливість інтеграції системи з принтерами різних виробників та конструкцій	На відміну від конкурента є підтримка альтернативних прошивок принтерів

Опис трьох рівнів моделі товару наведено у таблиці 5.19.

Таблиця 5.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
I. Товар за задумом	Товар дозволяє автоматизувати моніторинг процесу 3D друку, та автоматично зупинити друк при виявленні проблеми. Також товар надає можливість спостерігати за друком з будь-звідки та віддалено керувати принтером
II. Товар у реальному виконанні	Програмний продукт з відкритим сирцевим кодом що виконує функції,

	описані вище, веб сервіс, що дозволяє користувачу не перейматись за розгортання, налаштування продукту та забезпечення необхідних обчислювальних потужностей та мобільний додаток для взаємодії з продуктом
ІІІ. Товар з підкріпленням	До продажу: безкоштовний пробний період, безкоштовна версія, яку слід самостійно налаштовувати та розгортати
	Після продажу: підтримка користувачів
Товар буде захищатись від копіювання на рівні сирцевого коду за рахунок ліцензійного погодження, що забороняє комерційне використання.	

Визначення меж встановлення ціни наведено у таблиці 5.20.

Таблиця 5.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи користувачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	150 грн/місяць	150 грн/місяць	15000-40000 грн/місяць	130-250 грн/місяць для некомерційного вживання

				1000-2000 грн/місяць для комерційного вживання

Формування системи збуту наведено в таблиці 5.21.

Таблиця 5.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1.	Орієнтація на показник якості та зручності використання	Презентація переваг товару, реалізація продажу за підписковою моделлю	Напряму виробником	Дистрибуція через власний веб-сайт

У таблиці 5.22 наведена концепція маркетингових комунікацій.

Таблиця 5.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Основні канали комунікації у цільових груп клієнтів	Задачі реклами товару	Концепція рекламного повідомлення
1.	Орієнтація на показник якості та зручності використання	Меседжери, соціальні мережі, відеохостинги,	Показати якість послуг, що надаються, зручність та	Ролик з процесом використання продукту та

		форуми	переваги над конкурентами	демонстрацією його можливостей

Висновок розділу

З оглядом на проведене дослідження можна зробити наступні висновки:

– ринок 3D друку стрімко зростає та ніша моніторингу друку є практично не зайнятою;

– ринкова комерціалізація продукту має хороші шанси на успіх;

– існують перспективи впровадження з огляду на потенційні групи клієнтів, бар’єри входження не є високими, проект має дві значні переваги перед конкурентами;

– для ринкової реалізації проекту необхідно створити власне сам програмний продукт для моніторингу, веб сервіс, який власне і буде монетизуватись та мобільний додаток для взаємодії з продуктом;

– подальша імплементація є доцільною.

ВИСНОВКИ

У даній магістерській дисертації було вирішено проблему автоматизації моніторингу процесу 3D друку.Дана задача є актуально, оскільки, ця предметна область швидко розвивається.

В роботі розглянуті принцип роботи сучасних 3D принтерів та проаналізовано основні неполадки, що можуть виникати в процесі друку.

Запропоновано загальний принцип роботи системи для моніторингу процесу друку та розроблено три інноваційні методи для виявлення неполадок з допомогою аналізу візуальної інформації в контексті запропонованої системи, що не мають аналогів.

Кожен з розроблених методів було детально досліджено, виявлені їх переваги та недоліки. В роботі було зроблено огляд та аналіз технологій, що необхідні для реалізації даних методів у реальному програмному продукті.Вибір кожної з технології був обґрунтований. Оскільки система повинна взаємодіяти з 3D принтером та камерою, які повинні працювати у режимі реального часу, а робота з відео та нейронними мережами потребує велику кількість обчислювальнихресурсів, особливу увагу було надано розробці архітектури програмного забезпечення. Розроблено модульне рішення, що просто горизонтально масштабується, легко розширюється та є зручним у користуванні.

В роботі проведено аналіз ефективності роботи розроблених методів, що дозволило зробити висновок про надійність системи у цілому та доцільність використання зазначених підходів.

У рамках розробки стартап-проекту на основі створеного програмного забезпечення було проведено аналіз ринку, дослідження конкурентів, аналіз ринкових можливостей проекту, дослідження цільової аудиторії, створення ринкової та маркетингової стратегії.

Запропоновані в даній роботі інноваційні методи показали високу ефективність при застосуванні до процесу моніторингу 3D друку, що вперше дозволило повністю автоматизувати цей процес. В свою чергу, це дозволить суттєво спростити експлуатацію 3D принтерів та збільшить їх ефективність.

Створена система є не лише ефективною, готовою до практичного застосування, а й потенційно комерційноуспішною.

СПИСОК ДЖЕРЕЛ

1) Feeney D. 3D PrintingFailures: 2020 Edition: HowtoDiagnoseandRepair ALL Desktop 3D PrintingIssues / D. Feeney, A. Sean., 2020. – 540 с.

2) Johannes W. 3D Printing 101: TheUltimateBeginner'sGuide / WildJohannes., 2015. – 56 с.

3) Скворцов А. В. Алгоритмыпостроения и анализатриангуляции / СкворцовА.В., Мирза.Н.С. – Томск: Том, 2006. – 168 с.

4) Kadir G. Commonfdm 3d printingdefects / G. Kadir, T. Halit., 2018. – 42 с.

5) Alsoufi M. Warpingdeformationofdesktop 3d printedpartsmanufacturedbyopensourcefuseddepo-sitionmodeling (fdm) system. / M. Alsoufi, A. El-Sayed. // InternationalJournalofMechanical&MechatronicsEngineering. – 2017. – С.7-16.

6) AR S. Colorgamuttransformpairs, ComputerGraphics / Smith AR., 1978. – 103с.

7) Hartle R. Multipleviewgeometryincomputervision / R. Hartle, A. Zisserman. – Cambridgeuniversitypress, 2003. – 334 с.

8) Alex K. Imagenetclassificationwithdeepconvolutionalneuralnetworks / K. Alex, S. Ilya, H. GeoffreyHinton. – Pereira: CurranAssociates, Inc., 2012. – 1305 с.

9) WhatisMarlin? [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://marlinfw.org/.

10) Kessenich J. OpenGLProgrammingGuide: TheOfficialGuidetoLearningOpenGL / J. Kessenich, G. Sellers, D. Shreiner., 2017. – 1203 с. – (9).

11) Келер А. LearningOpenCV / А. Келер, Г. Брадський., 2008. – 403 с.

12) Shukla N. MachineLearningwithTensorFlow / NishantShukla., 2018. – 272 с.

13) Motion JPEG VideoAuthenticationbasedonQuantizationMatrixWatermarking: ApplicationinRobotics [Електронний ресурс] // InternationalJournalofComputerApplications. – 2012. –

Режим доступу до ресурсу: https://www.researchgate.net/publication/233793239_Motion_JPEG_Video_Authentication_based_on_Quantization_Matrix_Watermarking_Application_in_Robotics.

14) Grinberg M. FlaskWebDevelopment: DevelopingWebApplicationswithPython / MiguelGrinberg., 2018. – 310 с. – (2).

15) Macc M. REST API DesignRulebook / Марк Macc., 2011. – 114 с.

ДОДАТОК А ОПИС ПРОГРАМИ

mjpegsw.py

```
#!/usr/bin/python

importargparse
importtime
fromhttp.serverimportBaseHTTPRequestHandler, HTTPServer
fromioimportBytesIO
fromsocketserverimportThreadingMixIn

import cv2

from PIL importImage

capture = None


classCamHandler(BaseHTTPRequestHandler):
    # noinspection PyPep8Naming
defdo_GET(self):
ifself.path == '/favicon.ico':
return
print(f"{self.path}")
if '.mjpg' inself.path.lower():
            # sendvideostream
self.send_response(200)
self.send_header('Content-type', 'multipart/x-mixed-replace; boundary=--jpgboundary')
self.end_headers()
whileTrue:
try:
rc, img = capture.read()
ifnotrc:
continue
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
jpg = Image.fromarray(img_rgb)
tmp_file = BytesIO()
jpg.save(tmp_file, 'JPEG')
self.wfile.write(b"--jpgboundary")
self.end_headers()
self.send_header('Content-type', 'image/jpeg')
self.send_header('X-Timestamp', time.time())
self.send_header('Content-length', str(tmp_file.tell()))
self.end_headers()
tmp_file.seek(0)
self.wfile.write(tmp_file.read())
exceptKeyboardInterrupt:
break
except (BrokenPipeError, OSError):
pass
return
if '.jpg' inself.path.lower():
            # sendsnapshot
try:
rc, img = capture.read()
ifnotrc:
self.send_response(500)
return

img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
jpg = Image.fromarray(img_rgb)
```

```
tmp_file = BytesIO()
jpg.save(tmp_file, 'JPEG')

self.send_response(200)
self.send_header('Content-type', 'image/jpeg')
self.send_header('X-Timestamp', time.time())
self.send_header('Content-length', str(tmp_file.tell()))
self.end_headers()
tmp_file.seek(0)
self.wfile.write(tmp_file.read())
except (BrokenPipeError, OSError):
pass
return
ifself.path == '/':
self.send_response(200)
self.send_header('Content-type', 'text/html')
self.end_headers()
self.wfile.write(b'<html><head></head><body>')
self.wfile.write(b'<imgsrc="/cam.mjpg"/>')
self.wfile.write(b'</body></html>')
return


classThreadedHTTPServer(ThreadingMixIn, HTTPServer):
    """Handlerequestsin a separatethread."""


defhandle_args():
parser = argparse.ArgumentParser(description='Mjpegstreamingserver: mjpegsw -p 8080 --
camera 2')
parser.add_argument('-p', '--port', help='httplisteningport, default 5001', type=int,
default=5001)
parser.add_argument('-c', '--camera', help='opencvcameranumber, ex. -c 1', type=int,
default=0)
parser.add_argument('-i', '--ipaddress', help='listeningipaddress, defaultallips',
type=str,
default='0.0.0.0')
params = vars(parser.parse_args())
returnparams


defmain():
globalcapture

params = handle_args()
capture = cv2.VideoCapture(params['camera'])
server = ThreadedHTTPServer((params['ipaddress'], params['port']), CamHandler)

try:
print(f"Mjpegserverstartedon http://{params['ipaddress']}:{params['port']}")
server.serve_forever()
exceptKeyboardInterrupt:
capture.release()
server.socket.close()
exceptExceptionas e:
print(e)


if __name__ == '__main__':
main()
```

get_score.py

```
#!/usr/bin/env python

# import the necessary packages
from skimage.measure import compare_nrmse
from sys import argv, exit, stderr
from os.path import dirname
import cv2

# get filenames
fst_file = argv[1]
if fst_file[-4:] != ".jpg":
    exit()

# Set current image and previous image
curr = int(fst_file[-10:-4])
prev = curr - 1

# For the 1st and 2nd images, output 0
# 1st image is used as background image
# 2nd image has no previous image (gives errors as prev is BG)
if curr == 0:
    print("0 nan")
    print("Background Image", file=stderr)
    exit()
if curr == 1:
    print("1 nan")
    print("First Image, no previous images", file=stderr)
    exit()

snd_file = fst_file[:-10] + str(prev).rjust(6, '0') + fst_file[-4:]

bg_file = fst_file[:-10] + '000000.jpg'

# load the two input images and background image
#imageA = cv2.imread(fst_file)
#imageB = cv2.imread(snd_file)
#imageBG = cv2.imread(bg_file)

# convert the images to grayscale
grayA = cv2.imread(fst_file,0)
grayB = cv2.imread(snd_file,0)
grayBG = cv2.imread(bg_file,0)

# Remove background and threshold to remove shadow effects
threshold = 20

diffA = cv2.absdiff(grayA, grayBG)
thresA = cv2.threshold(diffA, threshold, 255, cv2.THRESH_BINARY)[1]

diffB = cv2.absdiff(grayB, grayBG)
thresB = cv2.threshold(diffB, threshold, 255, cv2.THRESH_BINARY)[1]

# compute the Normalised Root Mean-Squared Error (NRMSE) between the two
# images
score = compare_nrmse(thresA, thresB)

# Compare the current image with the image from 5 layers ago
# This is used to check for filament runout or huge deviance
deviance = 1.0
scr_diff = 0.0
dev_diff = 0.0
if curr > 5:
```

```
    trd_file = fst_file[:-10] + str(curr-5).rjust(6, '0') + fst_file[-4:]
    #imageC = cv2.imread(trd_file)
    grayC = cv2.imread(trd_file,0)
    diffC = cv2.absdiff(grayC, grayBG)
    thresC = cv2.threshold(diffC, threshold, 255, cv2.THRESH_BINARY)[1]
    deviance = compare_nrmse(thresA, thresC)

    # Calculate difference compared with previous layer score and deviance
    logfile = dirname(fst_file) + '/output.log'
    with open(logfile) as log:
        data = log.readlines()
    prev_layer = data[-1]
    layer,scr,dev,s_diff,d_diff = prev_layer.split(" ")
    scr_diff = abs(score-float(scr))
    dev_diff = abs(deviance-float(dev))

print("{} {} {} {} {}".format(curr,score,deviance,scr_diff,dev_diff))
print("Image: {:d}\t Score: {}\t Deviance: {}\tDiffs:
{}/{}".format(curr,score,deviance,scr_diff,dev_diff), file=stderr)
```

printcontrol.py

```
#!/usr/bin/env python

from sys import argv
from octoclient import OctoClient
from api_keys import URL, API_KEY

# Pause the printer
def pause_print():
    try:
        client = OctoClient(url=URL, apikey=API_KEY)
        flags = client.printer()['state']['flags']
        if flags['printing']:
client.pause()
print("Print paused.")
print("Layer: " + str(LAYER))
elif flags['paused'] or flags['pausing']:
print("Print already paused.")
        else:
print("Print cancelled or error occurred.")
    except Exception as e:
        print(e)


# Specify URL and API key for Octoprint
if URL is None:
    URL = 'YOUR OCTOPRINT IP ADDRESS'
if API_KEY is None:
    API_KEY = 'YOUR OCTOPRINT API KEY'


if argv[2] == 'nan': # Exit if SCORE is NaN, this occurs on the background and first layer
exit()

# Get SCORE, DEVIANCE and current layer from get_score.py

LAYER = int(argv[1])
# Do nothing if it is the background or first layer
if LAYER <= 7:
quit()
```

```
SCORE = float(argv[2])
DEVIANCE = float(argv[3])
SCR_DIFF = float(argv[4])
DEV_DIFF = float(argv[5])

# Detachment thresholds
SCR_THRES = 1.0
DEV_THRES = 1.0

# Partial Breakage thresholds for DIFF values
BR_SCR_THRES = 0.15
BR_DEV_THRES = 0.10

# Filament run out/clog thresholds
FIL_SCR_THRES = 0.23
FIL_DEV_THRES = 0.28


# This indicates the model has detached from the bed
if SCORE > SCR_THRES and DEVIANCE > DEV_THRES:
print("Cause: Print detached from bed")
pause_print()
# This indicates a part of the model has broken off
elif SCR_DIFF > BR_SCR_THRES and DEV_DIFF > BR_DEV_THRES:
print("Cause: Potential (partial) breakage")
pause_print()
elif SCORE < FIL_SCR_THRES and DEVIANCE < FIL_DEV_THRES:
print("Cause: Filament ran out or nozzle/extruder clog")
pause_print()
```

server.py

```
#!/usr/bin/env python

import flask
from flask import request, jsonify
from os import path, environ
from raven.contrib.flask import Sentry
import cv2
import numpy as np
import requests

from auth import token_required
from lib.detection_model import load_net, detect

THRESH = 0.08  # The threshold for a box to be considered a positive detection
SESSION_TTL_SECONDS = 60*2

app = flask.Flask(__name__)

# SECURITY WARNING: don't run with debug turned on in production!
app.config['DEBUG'] = environ.get('DEBUG') == 'True'

# Sentry
sentry = None
if environ.get('SENTRY_DSN'):
    sentry = Sentry(app, dsn=environ.get('SENTRY_DSN'))

model_dir = path.join(path.dirname(path.realpath(__file__)), 'model')
```

```
net_main, meta_main = load_net(path.join(model_dir, 'model.cfg'), path.join(model_dir,
'model.weights'), path.join(model_dir, 'model.meta'))

@app.route('/p/', methods=['GET'])
@token_required
def get_p():
    if 'img' in request.args:
        try:
            resp = requests.get(request.args['img'], stream=True, timeout=(0.1, 5))
resp.raise_for_status()
img_array = np.array(bytearray(resp.content), dtype=np.uint8)
img = cv2.imdecode(img_array, -1)
            detections = detect(net_main, meta_main, img, thresh=THRESH)
            return jsonify({'detections': detections})
        except:
            if sentry:
sentry.captureException()
    else:
app.logger.warn("Invalid request params: {}".format(request.args))

    return jsonify({'detections': []})

if __name__ == "__main__":
app.run(host='0.0.0.0', port=3333, threaded=False)

import glfw
from OpenGL.GL import *
import OpenGL.GL.shaders
import numpy as np
import pyrr
from PIL import Image


def main():
    if not glfw.init():
        return

    if not window:
glfw.terminate()
        return

glfw.make_context_current(window)

    cube = np.array(cube, dtype=np.float32)

    indices = np.array(indices, dtype=np.uint32)

    VERTEX_SHADER = """

              #version 330

              in vec3 position;
              in vec3 color;
              in vec2 InTexCoords;

              out vec3 newColor;
              out vec2 OutTexCoords;

              uniform mat4 transform;


              uniform mat4 view;
```

```
              uniform mat4 model;
              uniform mat4 projection;

              void main() {

gl_Position = projection * view * model * transform * vec4(position, 1.0f);
newColor = color;
OutTexCoords = InTexCoords;

                }


          """

    FRAGMENT_SHADER = """
           #version 330

            in vec3 newColor;
            in vec2 OutTexCoords;

            out vec4 outColor;
            uniform sampler2D samplerTex;

           void main() {

outColor = texture(samplerTex, OutTexCoords);

           }

       """

    shader =
OpenGL.GL.shaders.compileProgram(OpenGL.GL.shaders.compileShader(VERTEX_SHADER,
GL_VERTEX_SHADER),
OpenGL.GL.shaders.compileShader(FRAGMENT_SHADER, GL_FRAGMENT_SHADER))


    VBO = glGenBuffers(1)
glBindBuffer(GL_ARRAY_BUFFER, VBO)
glBufferData(GL_ARRAY_BUFFER, cube.itemsize * len(cube), cube, GL_STATIC_DRAW)


    # Create EBO
    EBO = glGenBuffers(1)
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, EBO)
glBufferData(GL_ELEMENT_ARRAY_BUFFER, indices.itemsize * len(indices), indices,
GL_STATIC_DRAW)


    # get the position from  shader
    position = glGetAttribLocation(shader, 'position')
glVertexAttribPointer(position, 3, GL_FLOAT, GL_FALSE, cube.itemsize * 8,
ctypes.c_void_p(0))
glEnableVertexAttribArray(position)


    color = glGetAttribLocation(shader, 'color')
glVertexAttribPointer(color, 3, GL_FLOAT, GL_FALSE, cube.itemsize * 8,
ctypes.c_void_p(12))
glEnableVertexAttribArray(color)
```

```
texCoords = glGetAttribLocation(shader, "InTexCoords")
glVertexAttribPointer(texCoords, 2, GL_FLOAT, GL_FALSE, cube.itemsize * 8,
ctypes.c_void_p(24))
glEnableVertexAttribArray(texCoords)

    texture = glGenTextures(1)
glBindTexture(GL_TEXTURE_2D, texture)


    # Set the texture wrapping parameters
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT)
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT)


glUseProgram(shader)

glClearColor(0.0, 0.0, 0.0, 1.0)
glEnable(GL_DEPTH_TEST)


    #Creating Projection Matrix
    view =pyrr.matrix44.create_from_translation(pyrr.Vector3([0.0,0.0,-3.0] ))
    projection = pyrr.matrix44.create_perspective_projection(20.0, 720/600, 0.1, 100.0)
    model = pyrr.matrix44.create_from_translation(pyrr.Vector3([0.0,0.0,0.0]))

view_loc = glGetUniformLocation(shader, "view")
proj_loc = glGetUniformLocation(shader, "projection")
model_loc = glGetUniformLocation(shader, "model")

    glUniformMatrix4fv(view_loc, 1, GL_FALSE, view)
    glUniformMatrix4fv(proj_loc, 1, GL_FALSE, projection)
    glUniformMatrix4fv(model_loc, 1, GL_FALSE, model)


    while not glfw.window_should_close(window):
glfw.poll_events()

glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

rot_x = pyrr.Matrix44.from_x_rotation(0.5 * glfw.get_time())
rot_y = pyrr.Matrix44.from_y_rotation(0.8 * glfw.get_time())

transformLoc = glGetUniformLocation(shader, "transform")
        glUniformMatrix4fv(transformLoc, 1, GL_FALSE, rot_x * rot_y)

        # Draw Cube

glDrawElements(GL_TRIANGLES, 36, GL_UNSIGNED_INT, None)

glfw.swap_buffers(window)

glfw.terminate()
```

model.py

```
from allauth.account.admin import EmailAddress
from datetime import datetime, timedelta
import logging
import os
import jsonfromjsonfield import JSONField
import uuid
from simple_history.models import HistoricalRecords
```

```
from safedelete.models import SafeDeleteModel
from safedelete.managers import SafeDeleteManager
from pushbullet import Pushbullet, errors
from django.utils.html import mark_safe

from config.celery import celery_app
from lib import cache, channels
from lib.utils import dict_or_none

LOGGER = logging.getLogger(__name__)

UNLIMITED_DH =


def dh_is_unlimited(dh):
    return dh >= UNLIMITED_DH


class UserManager(BaseUserManager):

use_in_migrations = True

class User(AbstractUser):
    username = None
    email = models.EmailField(_('email address'), unique=True)
phone_number = models.CharField(max_length=20, null=True, blank=True)
phone_country_code = models.CharField(max_length=5, null=True, blank=True)
pushbullet_access_token = models.CharField(max_length=45, null=True, blank=True)
telegram_chat_id = models.BigIntegerField(null=True, blank=True)
slack_access_token = models.CharField(max_length=128, null=True, blank=True)
consented_at = models.DateTimeField(null=True, blank=True)
is_pro = models.BooleanField(null=False, blank=False, default=True)
dh_balance = models.FloatField(null=False, default=0)
unsub_token = models.UUIDField(null=False, blank=False, unique=True, db_index=True,
default=uuid.uuid4, editable=False)
notify_on_done = models.BooleanField(null=False, blank=False, default=True)
notify_on_canceled = models.BooleanField(null=False, blank=False, default=False)
print_notification_by_email = models.BooleanField(null=False, blank=False, default=True)
account_notification_by_email = models.BooleanField(null=False, blank=False, default=True)
alert_by_email = models.BooleanField(null=False, blank=False, default=True)
print_notification_by_pushbullet = models.BooleanField(null=False, blank=False,
default=True)
print_notification_by_telegram = models.BooleanField(null=False, blank=False,
default=True)
alert_by_sms = models.BooleanField(null=False, blank=False, default=True)
discord_webhook = models.CharField(max_length=256, null=True, blank=True)
print_notification_by_discord = models.BooleanField(null=False, blank=False, default=True)

    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = []

    objects = UserManager()

    def sms_eligible(self):
        return self.phone_number and self.phone_country_code

    def is_primary_email_verified(self):
        if EmailAddress.objects.filter(user=self, email=self.email,
                                       verified=True).exists():

            return True
```

```
        return False

    def has_verified_email(self):
        # Give user 1 day before bugging them to verify their email addresses

        return timezone.now() - timedelta(days=1) <self.date_joined or
EmailAddress.objects.filter(user=self, verified=True).exists()

    def has_valid_pushbullet_token(self):
        if not self.pushbullet_access_token:
            return False

        try:
Pushbullet(self.pushbullet_access_token)

            return True
        except errors.InvalidKeyError:
            return False

    def tunnel_usage_over_cap(self):
        return not self.is_pro and cache.octoprinttunnel_get_stats(self.id)
>settings.OCTOPRINT_TUNNEL_CAP * 1.1  # Cap x 1.1 to give some grace period to users

# We use a signal as opposed to a form field because users may sign up using social
buttons


@receiver(post_save, sender=User)
def update_consented_at(sender, instance, created, **kwargs):
    if created:
instance.consented_at = timezone.now()
instance.save()


class PrinterManager(SafeDeleteManager):
    def get_queryset(self):
        return super(PrinterManager, self).get_queryset().filter(archived_at__isnull=True)


class Printer(SafeDeleteModel):
    class Meta:
default_manager_name = 'objects'

    PAUSE = 'PAUSE'
    NONE = 'NONE'
    ACTION_ON_FAILURE = (
        (NONE, 'Just notify me'),
        (PAUSE, 'Pause the printer and notify me'),
    )

    name = models.CharField(max_length=200, null=False)
auth_token = models.CharField(max_length=28, unique=True, null=False, blank=False)
    user = models.ForeignKey(User, on_delete=models.CASCADE, null=False)
current_print = models.OneToOneField('Print', on_delete=models.SET_NULL, null=True,
blank=True, related_name='not_used')
action_on_failure = models.CharField(
max_length=10,
        choices=ACTION_ON_FAILURE,
        default=PAUSE,
    )
watching_enabled = models.BooleanField(default=True, db_column="watching")
tools_off_on_pause = models.BooleanField(default=True)
```

```
bed_off_on_pause = models.BooleanField(default=False)
retract_on_pause = models.FloatField(null=False, default=6.5)
lift_z_on_pause = models.FloatField(null=False, default=2.5)
detective_sensitivity = models.FloatField(null=False, default=1.0)

archived_at = models.DateTimeField(null=True, blank=True)
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)
service_token = models.CharField(max_length=64, unique=True, db_index=True, null=True,
blank=False)

    objects = PrinterManager()
with_archived = SafeDeleteManager()

    if os.environ.get('ENALBE_HISTORY', '') == 'True':
        history = HistoricalRecords(excluded_fields=['updated_at'])

    @property
    def status(self):
status_data = cache.printer_status_get(self.id)

        for k, v in status_data.items():
status_data[k] = json.loads(v)

        return dict_or_none(status_data)

    @property
    def pic(self):
pic_data = cache.printer_pic_get(self.id)

        return dict_or_none(pic_data)

    @property
    def settings(self):
p_settings = cache.printer_settings_get(self.id)

        for key in ('webcam_flipV', 'webcam_flipH', 'webcam_rotate90'):
p_settings[key] = p_settings.get(key, 'False') == 'True'
p_settings['ratio169'] = p_settings.get('webcam_streamRatio', '4:3') == '16:9'

        if p_settings.get('temp_profiles'):
p_settings['temp_profiles'] = json.loads(p_settings.get('temp_profiles'))

        return p_settings

    def should_watch(self):
        if not self.watching_enabled or self.user.dh_balance< 0:
            return False

        return self.current_print is not None and self.current_print.alert_muted_at is
None

    def not_watching_reason(self):
        if not self.watching_enabled:
            return '"Watch for failures" is turned off'

        if self.user.dh_balance< 0:
            return "You have ran out of Detective Hours"

        if not self.actively_printing():
            return "Printer is not actively printing"
```

```
        if self.current_print is not None and self.current_print.alert_muted_at is not
None:
            return "Alerts are muted for current print"

        return None

    def actively_printing(self):
printer_cur_state = cache.printer_status_get(self.id, 'state')

        return printer_cur_state and json.loads(printer_cur_state).get('flags',
{}).get('printing', False)

    def update_current_print(self, filename, current_print_ts):
        if current_print_ts == -1
            if self.current_print:
                if self.current_print.started_at< (timezone.now() - timedelta(hours=10)):
self.unset_current_print()
                else:
LOGGER.warn(f'current_print_ts=-1 received when current print is still active. print_id:
{self.current_print_id} - printer_id: {self.id}')

            return

        # currently printing

        if self.current_print:
            if self.current_print.ext_id == current_print_ts:
                return

            if self.current_print.ext_id in range(current_print_ts - 20, current_print_ts
+ 20) and self.current_print.filename == filename:
LOGGER.warn(
f'Apparently skewed print_ts received. ts1: {self.current_print.ext_id} - ts2:
{current_print_ts} - print_id: {self.current_print_id} - printer_id: {self.id}')

                return
LOGGER.warn(f'Print not properly ended before next start. Stale print_id:
{self.current_print_id} - printer_id: {self.id}')
self.unset_current_print()
self.set_current_print(filename, current_print_ts)
        else:
self.set_current_print(filename, current_print_ts)

    def unset_current_print(self):
        print = self.current_print
self.current_print = None
self.save()

self.printerprediction.reset_for_new_print()

        if print.cancelled_at is None:
print.finished_at = timezone.now()
print.save()

PrintEvent.create(print, PrintEvent.ENDED)
self.send_should_watch_status()

    def set_current_print(self, filename, current_print_ts):
        if not current_print_ts or current_print_ts == -1:
            raise Exception(f'Invalidcurrent_print_ts when trying to set current_print:
{current_print_ts}')
```

```
cur_print, _ = Print.objects.get_or_create(
            user=self.user,
            printer=self,
ext_id=current_print_ts,
            defaults={'filename': filename, 'started_at': timezone.now()},
        )

        if cur_print.ended_at():
            if cur_print.ended_at() > (timezone.now() - timedelta(seconds=30)):  # Race
condition. Some msg with valid print_ts arrived after msg with print_ts=-1
                return
            else:
                raise Exception('Ended print is re-surrected! printer_id: {} | print_ts:
{} | filename: {}'.format(self.id, current_print_ts, filename))

self.current_print = cur_print
self.save()

self.printerprediction.reset_for_new_print()
PrintEvent.create(cur_print, PrintEvent.STARTED)
self.send_should_watch_status()

    ## return: succeeded? ##
    def resume_print(self, mute_alert=False):
        if self.current_print is None:  # when a link on an old email is clicked
            return False

self.current_print.paused_at = None
self.current_print.save()

self.acknowledge_alert(Print.NOT_FAILED)
self.send_octoprint_command('resume')

        return True

    ## return: succeeded? ##
    def pause_print(self):
        if self.current_print is None:
            return False

self.current_print.paused_at = timezone.now()
self.current_print.save()

args = {'retract': self.retract_on_pause, 'lift_z': self.lift_z_on_pause}

        if self.tools_off_on_pause:
args['tools_off'] = True

        if self.bed_off_on_pause:
args['bed_off'] = True
self.send_octoprint_command('pause', args=args)

        return True

    ## return: succeeded? ##
    def cancel_print(self):
        if self.current_print is None
            return False

self.acknowledge_alert(Print.FAILED)
self.send_octoprint_command('cancel')
```

```
        return True

    def set_alert(self):
self.current_print.alerted_at = timezone.now()
self.current_print.save()

    def acknowledge_alert(self, alert_overwrite):
        if not self.current_print.alerted_at:
            return

self.current_print.alert_acknowledged_at = timezone.now()
self.current_print.alert_overwrite = alert_overwrite
self.current_print.save()

    def mute_current_print(self, muted):
self.current_print.alert_muted_at = timezone.now() if muted else None
self.current_print.save()

        if muted:
PrintEvent.create(self.current_print, PrintEvent.ALERT_MUTED)
        else:
PrintEvent.create(self.current_print, PrintEvent.ALERT_UNMUTED)

self.send_should_watch_status()

    # messages to printer

    def send_octoprint_command(self, command, args={}):
channels.send_msg_to_printer(self.id, {'commands': [{'cmd': command, 'args': args}]})

    def send_should_watch_status(self):
self.refresh_from_db()
channels.send_msg_to_printer(self.id, {'remote_status': {'should_watch':
self.should_watch()}})

    def __str__(self):
        return str(self.id)


class HeaterTracker(models.Model):
    class Meta:
unique_together = ('printer', 'name')

    printer = models.ForeignKey(Printer, on_delete=models.CASCADE)
    name = models.CharField(max_length=16, blank=False)
    target = models.FloatField()
    reached = models.BooleanField()

created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)


class PrinterCommand(models.Model):
    PENDING = 'PENDING'
    SENT = 'SENT'
    ABORTED = 'ABORTED'

    COMMAND_STATUSES = (
        (PENDING, 'pending'),
        (SENT, 'sent'),
        (ABORTED, 'aborted'),
    )
```

```
    printer = models.ForeignKey(Printer, on_delete=models.CASCADE, null=False)
    command = models.CharField(max_length=2000, null=False, blank=False)
    status = models.CharField(
max_length=10,
        choices=COMMAND_STATUSES,
        default=PENDING,
    )
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)


def calc_normalized_p(detective_sensitivity: float,
pred: 'PrinterPrediction') -> float:
    def scale(oldValue, oldMin, oldMax, newMin, newMax):
newValue = (((oldValue - oldMin) * (newMax - newMin)) / (oldMax - oldMin)) + newMin
        return min(newMax, max(newMin, newValue))

thresh_warning = (pred.rolling_mean_short - pred.rolling_mean_long) *
settings.ROLLING_MEAN_SHORT_MULTIPLE
thresh_warning = min(settings.THRESHOLD_HIGH, max(settings.THRESHOLD_LOW, thresh_warning))
thresh_failure = thresh_warning * settings.ESCALATING_FACTOR

    p = (pred.ewm_mean - pred.rolling_mean_long) * detective_sensitivity

    if p >thresh_failure:
        return scale(p, thresh_failure, thresh_failure * 1.5, 2.0 / 3.0, 1.0)
elif p >thresh_warning:
        return scale(p, thresh_warning, thresh_failure, 1.0 / 3.0, 2.0 / 3.0)
    else:
        return scale(p, 0, thresh_warning, 0, 1.0 / 3.0)


class PrinterPrediction(models.Model):
    printer = models.OneToOneField(Printer, on_delete=models.CASCADE, primary_key=True)
current_frame_num = models.IntegerField(null=False, default=0)
lifetime_frame_num = models.IntegerField(null=False, default=0)
current_p = models.FloatField(null=False, default=0.0)
ewm_mean = models.FloatField(null=False, default=0.0)
rolling_mean_long = models.FloatField(null=False, default=0.0)
rolling_mean_short = models.FloatField(null=False, default=0.0)
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)

    def reset_for_new_print(self):
self.current_frame_num = 0
self.current_p = 0.0
self.ewm_mean = 0.0
self.rolling_mean_short = 0.0
self.save()

    def __str__(self):
        return '| printer_id: {} | current_p: {:.4f} | ewm_mean: {:.4f} |
rolling_mean_short: {:.4f} | rolling_mean_long: {:.4f} | current_frame_num: {} |
lifetime_frame_num: {} |'.format(
self.printer_id,
self.current_p,
self.ewm_mean,
self.rolling_mean_short,
self.rolling_mean_long,
self.current_frame_num,
self.lifetime_frame_num,
```

```
        )


@receiver(post_save, sender=Printer)
def create_printer_prediction(sender, instance, created, **kwargs):
    if created:
PrinterPrediction.objects.create(printer=instance)


class PublicTimelapse(models.Model):
    title = models.CharField(max_length=500)
    priority = models.IntegerField(null=False, default=1000000)
video_url = models.CharField(max_length=2000, null=False, blank=False)
poster_url = models.CharField(max_length=2000, null=False, blank=False)
p_json_url = models.CharField(max_length=2000, null=False, blank=False)
creator_name = models.CharField(max_length=500, null=False, blank=False)
uploaded_by = models.ForeignKey(User, on_delete=models.SET_NULL, null=True)
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)


class Print(SafeDeleteModel):

    class Meta:
unique_together = [['printer', 'ext_id']]

    FAILED = 'FAILED'
    NOT_FAILED = 'NOT_FAILED'
    PARTIALY_FAILED = 'PARTIALY_FAILED'

    ALERT_OVERWRITE = (
        (FAILED, FAILED),
        (NOT_FAILED, NOT_FAILED),
        (PARTIALY_FAILED, PARTIALY_FAILED),
    )

    printer = models.ForeignKey(Printer, on_delete=models.CASCADE, null=True)
    user = models.ForeignKey(User, on_delete=models.CASCADE, null=False)
ext_id = models.IntegerField(null=True, blank=True)
    filename = models.CharField(max_length=1000, null=False, blank=False)
started_at = models.DateTimeField(null=True)
finished_at = models.DateTimeField(null=True)
cancelled_at = models.DateTimeField(null=True)
uploaded_at = models.DateTimeField(null=True)
alerted_at = models.DateTimeField(null=True)
alert_acknowledged_at = models.DateTimeField(null=True)
alert_muted_at = models.DateTimeField(null=True)
paused_at = models.DateTimeField(null=True)
video_url = models.CharField(max_length=2000, null=True)
tagged_video_url = models.CharField(max_length=2000, null=True)
poster_url = models.CharField(max_length=2000, null=True)
prediction_json_url = models.CharField(max_length=2000, db_index=True, null=True)
alert_overwrite = models.CharField(
max_length=20,
        choices=ALERT_OVERWRITE,
        null=True
    )
access_consented_at = models.DateTimeField(null=True, blank=True)
created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)

    if os.environ.get('ENALBE_HISTORY', '') == 'True':
```

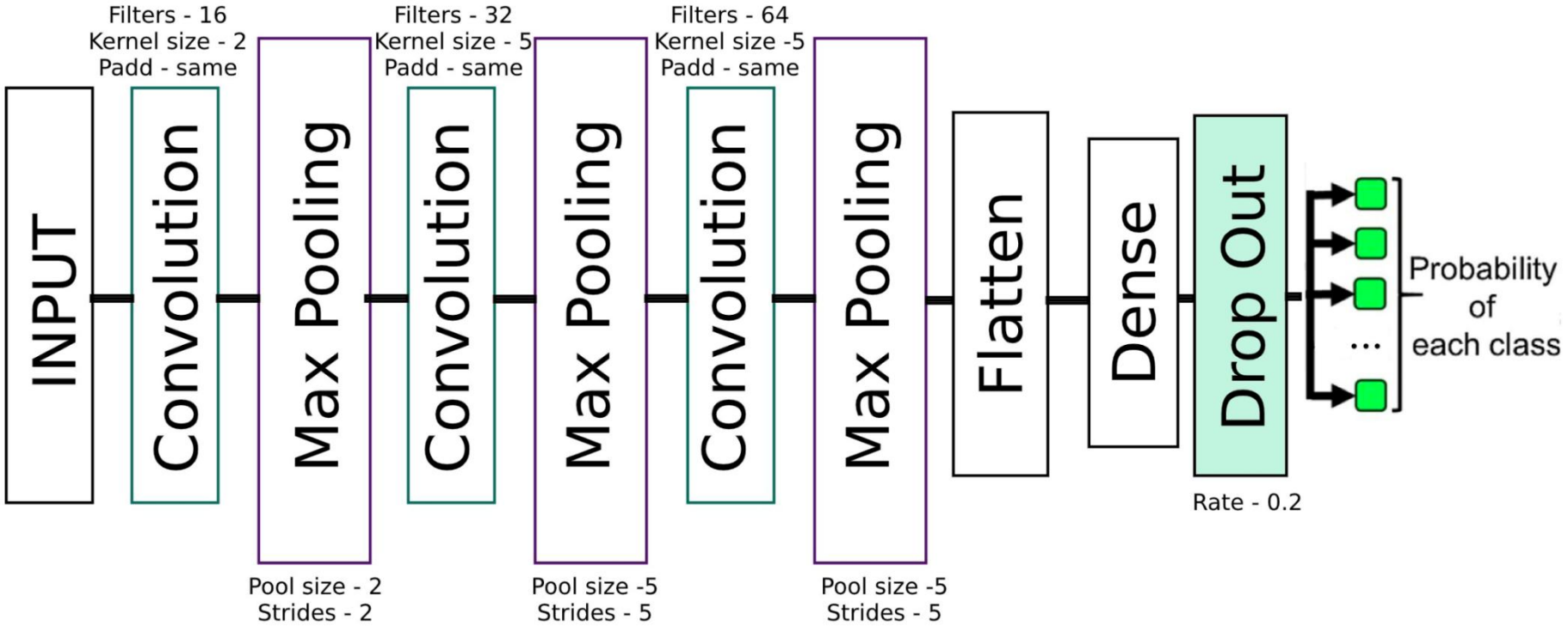
```
        history = HistoricalRecords(excluded_fields=['updated_at'])

    def ended_at(self):
        return self.cancelled_at or self.finished_at

    def duration(self):
        return self.ended_at() - self.started_at

    def has_alerted(self):
        return self.alerted_at

    def is_canceled(self):
        return bool(self.cancelled_at)

    @property
    def expecting_detective_view(self):
        return self.tagged_video_url or self.uploaded_at


class PrintEvent(models.Model):
    STARTED = 'STARTED'
    ENDED = 'ENDED'
    PAUSED = 'PAUSED'
    RESUMED = 'RESUMED'
    ALERT_MUTED = 'ALERT_MUTED'
    ALERT_UNMUTED = 'ALERT_UNMUTED'

    EVENT_TYPE = (
        (STARTED, STARTED),
        (ENDED, ENDED),
        (PAUSED, PAUSED),
        (RESUMED, RESUMED),
        (ALERT_MUTED, ALERT_MUTED),
        (ALERT_UNMUTED, ALERT_UNMUTED),
    )

    STOPPING_EVENT_TYPES = (ENDED, PAUSED, ALERT_MUTED)

    print = models.ForeignKey(Print, on_delete=models.CASCADE, null=False)
event_type = models.CharField(
max_length=20,
        choices=EVENT_TYPE,
        null=True
    )
alert_muted = models.BooleanField(null=False)
created_at = models.DateTimeField(auto_now_add=True)

    def create(print, event_type):
        event = PrintEvent.objects.create(
            print=print,
event_type=event_type,
alert_muted=(print.alert_muted_at is not None)
        )

        if event_type in PrintEvent.ENDED:
celery_app.send_task(settings.PRINT_EVENT_HANDLER, args=[print.id])


class SharedResource(models.Model):
    printer = models.OneToOneField(Printer, on_delete=models.CASCADE, null=True)
share_token = models.CharField(max_length=40, unique=True, null=False, blank=False)
```

```
created_at = models.DateTimeField(auto_now_add=True, db_index=True)
updated_at = models.DateTimeField(auto_now=True)


class GCodeFile(SafeDeleteModel):
    user = models.ForeignKey(User, on_delete=models.CASCADE, null=False)
    filename = models.CharField(max_length=1000, null=False, blank=False)
safe_filename = models.CharField(max_length=1000, null=False, blank=False)
url = models.CharField(max_length=2000, null=False, blank=False)
num_bytes = models.BigIntegerField(null=True, blank=True)

created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)


class PrintShotFeedback(models.Model):
    LOOKS_BAD = 'LOOKS_BAD'
    LOOKS_OK = 'LOOKS_OK'
    UNANSWERED = 'UNDECIDED'

    ANSWER_CHOICES = (
        (LOOKS_BAD, "It contains spaghetti"),
        (LOOKS_OK, "It does NOT contain spaghetti"),
        (UNANSWERED, "I'll decide later"),
    )

    print = models.ForeignKey(Print, on_delete=models.CASCADE)

image_url = models.CharField(max_length=2000, null=False, blank=False)

    answer = models.CharField(max_length=16, choices=ANSWER_CHOICES, blank=True,
null=True, db_index=True)
answered_at = models.DateTimeField(null=True, blank=True, db_index=True)
persisted_at = models.DateTimeField(null=True, blank=True, db_index=True)

created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)

    def image_tag(self):
        return mark_safe(f'<imgsrc="{self.image_url}" width="150" height="150" />')

image_tag.short_description = 'Image'

class ActiveMobileDeviceManager(models.Manager):
    def get_queryset(self):
        return super(ActiveMobileDeviceManager,
self).get_queryset().filter(deactivated_at__isnull=True)

class MobileDevice(models.Model):

    class Meta:
unique_together = [['user', 'device_token']]

    user = models.ForeignKey(User, on_delete=models.CASCADE, null=False)
    platform = models.CharField(max_length=16, null=False, blank=False)
app_version = models.CharField(max_length=16, null=False, blank=False)
device_token = models.CharField(max_length=256, null=False, blank=False)
deactivated_at = models.DateTimeField(null=True, blank=True, db_index=True)
preferred_timezone = models.CharField(max_length=32, null=True, blank=False)

created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)
```

```
    objects = ActiveMobileDeviceManager()
with_inactive = models.Manager()
```

ДОДАТОК Б ГРАФІЧНИЙ МАТЕРІАЛ

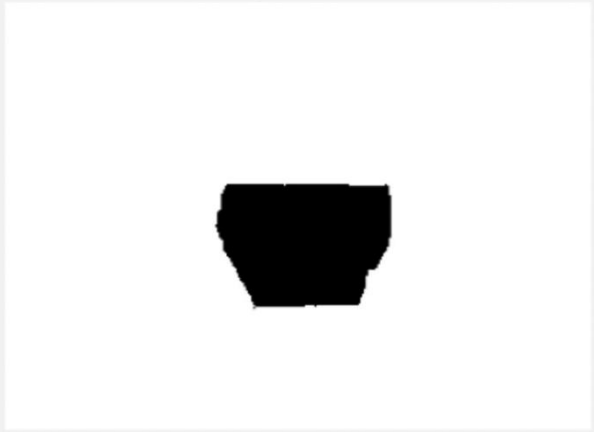
АРХІТЕКТУРА НЕЙРОННОЇ МЕРЕЖІ



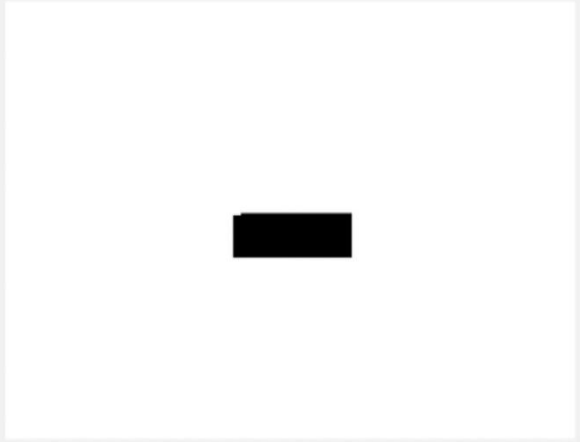
РЕЗУЛЬТАТ СЕГМЕНТАЦІЇ ТА ГЕНЕРАЦІЇ ЗОБРАЖЕННЯ

Зображення після сегментації

Зображення після сегментації



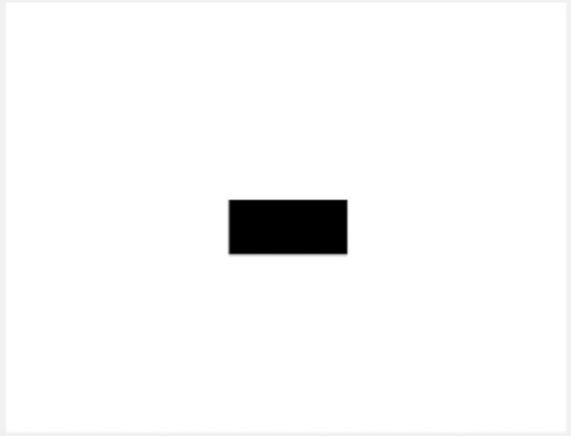
Зображення після сегментації



Генероване зображення

Генероване зображення

Генероване зображення



UNICHECK

Ім'я користувача:
Попенко Володимир Дмитрович

Дата перевірки:
15.12.2020 07:23:17 EET

Дата звіту:
15.12.2020 07:54:54 EET

ID перевірки:
1005458560

Тип перевірки:
Doc vs Internet + Library

ID користувача:
77149

Назва документа: Jadelskij_magistr_ip92mp_2

Кількість сторінок: 66 **Кількість слів:** 11124 **Кількість символів:** 85655 **Розмір файлу:** 17.36 MB **ID файлу:** 1005748485

11.1%
Схожість

Найбільша схожість: 1.17% з Інтернет-джерелом (https://cad.kpi.ua/attachments/093_2017dm_myronenko.pdf)

6.67% Джерела з Інтернету | 301 ..... Сторінка 68

8.58% Джерела з Бібліотеки | 557 ..... Сторінка 71

0% Цитат

Не знайдено жодних цитат

Вилучення списку бібліографічних посилань вимкнене

88.8%
Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 8 слів та 0%)

Немає вилучених Інтернет-джерел

88.8% Вилученого тексту з Бібліотеки | 1 ..... Сторінка 71

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | 3